Decomposing Terminating Rewrite Relations

Jörg Endrullis [1] Dieter Hofbauer [2] Johannes Waldmann [3]

[1]Department of Computer Science, Vrije Universiteit Amsterdam
De Boelelaan 1081, 1081 HV Amsterdam, The Netherlands.
Email: joerg@few.vu.nl

[2]Mühlengasse 16, D-34125 Kassel, Germany
Email: dieter@theory.informatik.uni-kassel.de

[3]Hochschule für Technik, Wirtschaft und Kultur (FH) Leipzig
Fb IMN, PF 30 11 66, D-04251 Leipzig, Germany.
Email: waldmann@imn.htwk-leipzig.de

Motivation and Outline

Remark (Motivation)

Construction of certificate automaton for match-boundedness. Up to now

- *exact but inefficient algorithm [Matchbox03]*
- *approximation, fast but incomplete [Torpa04,Aprove05,TTTBox06]*

This talk explains the algorithm of [Jambox05, Matchbox06]

Decomposition

We use the following classes of string rewriting systems (SRS):

- $\mathrm{CF} = \{R \mid \forall(\ell \to r) \in R : |\ell| \leq 1\}$ (context-free SRS)
- $\mathrm{CF}_0 = \{R \mid \forall(\ell \to r) \in R : |\ell| = 0\} \subseteq \mathrm{CF}$
- $\mathrm{SN} = \{R \mid \to_R$ is strongly normalizing (terminating)$\}$

We write R^- for $\{r \to \ell \mid (\ell \to r) \in R\}$.

Definition (Decomposition of R)

Let R be a SRS over Σ, let S and T be SRSs over $\Gamma \supseteq \Sigma$.
Then the pair (S, T) is a decomposition of R if

$$\to_R^* = (\to_S^* \circ \to_T^*) \cap (\Sigma^* \times \Sigma^*).$$

If additionally $S \in \mathcal{S}$ and $T \in \mathcal{T}$ for classes of SRSs $\mathcal{S}$ and $\mathcal{T}$,
then (S, T) is called an $(\mathcal{S}, \mathcal{T})$-decomposition of R.

Left and right inverse letters

$(\Sigma^*, \cdot)$ is a monoid, but concatenation is not invertible.
We introduce formal left and right inverses of letters:

- $\overline{\Sigma} = \Sigma \uplus \{\, \overrightarrow{a},\ \overleftarrow{a} \mid a \in \Sigma\}$

The behaviour of these is expressed by the rewriting system:

- $E = \{\, \overrightarrow{a}\, a \to \epsilon,\ a\, \overleftarrow{a} \to \epsilon \mid a \in \Sigma\}$

We extend $^{\rightarrow}$ and $^{\leftarrow}$ from letters to strings by:

- $\overrightarrow{a_1 \cdots a_n} = \overrightarrow{a_n} \cdots \overrightarrow{a_1}$ and $\overleftarrow{a_1 \cdots a_n} = \overleftarrow{a_n} \cdots \overleftarrow{a_1}$

Observe that $\overrightarrow{x}\, x \to_E^* \epsilon \leftarrow_E^* x\, \overleftarrow{x}$ for $x \in \Sigma^*$.

The above construction is standard. The congruence relation generated by $\to_E$ is called the Shamir congruence in [Sak03] II.6.2.

Conjugates

Example

The system $R = \{ab \to c\}$ over $\Sigma = \{a, b, c\}$ has the conjugates
R, $\{b \to \overleftarrow{a}\, c\}$, $\{a \to c\, \overrightarrow{b}\}$, $\{\epsilon \to \overleftarrow{a}\, c\, \overrightarrow{b}\}$, $\{\epsilon \to \overleftarrow{b}\, \overleftarrow{a}\, c\}$, $\{\epsilon \to c\, \overrightarrow{b}\, \overrightarrow{a}\}$.

By $C(R)$ we denote the union of all conjugates of R.

Lemma

For every SRS R over Σ,

(1) $\to^*_{C(R) \cup E} \cap (\Sigma^* \times \Sigma^*) \subseteq \to^*_R$ *(correctness)*

(2) $\to^*_R \subseteq \to^*_C \circ \to^*_E$, *(completeness)*

for every context-free conjugate C of R

Theorem

Every SRS R has a context-free conjugate C and

- (C, E) *is a* $(\mathrm{CF}, \mathrm{SN} \cap \mathrm{CF}_0^-)$*-decomposition of R*
- $(C(R), E)$ *is a decomposition of R*

Conjugates

Example

The system $R = \{ab \to c\}$ over $\Sigma = \{a, b, c\}$ has the conjugates
R, $\{b \to \overleftarrow{a}\, c\}$, $\{a \to c\, \overrightarrow{b}\}$, $\{\epsilon \to \overleftarrow{a}\, c\, \overrightarrow{b}\}$, $\{\epsilon \to \overleftarrow{b}\, \overleftarrow{a}\, c\}$, $\{\epsilon \to c\, \overrightarrow{b}\, \overrightarrow{a}\}$.

By $C(R)$ we denote the union of all conjugates of R.

Lemma

For every SRS R over Σ,

(1) $\to^*_{C(R)\cup E} \cap (\Sigma^* \times \Sigma^*) \subseteq \to^*_R$ *(correctness)*

(2) $\to^*_R \subseteq \to^*_C \circ \to^*_E$, *(completeness)*
for every context-free conjugate C of R

Theorem

Every SRS R has a context-free conjugate C and

- (C, E) *is a* $(\mathrm{CF}, \mathrm{SN} \cap \mathrm{CF}_0^-)$*-decomposition of R*
- $(C(R), E)$ *is a decomposition of R*

Conjugates

Example

The system $R = \{ab \to c\}$ over $\Sigma = \{a, b, c\}$ has the conjugates
R, $\{b \to \overleftarrow{a}\, c\}$, $\{a \to c\, \overrightarrow{b}\}$, $\{\epsilon \to \overleftarrow{a}\, c\, \overrightarrow{b}\}$, $\{\epsilon \to \overleftarrow{b}\, \overleftarrow{a}\, c\}$, $\{\epsilon \to c\, \overrightarrow{b}\, \overrightarrow{a}\}$.

By $C(R)$ we denote the union of all conjugates of R.

Lemma

For every SRS R over Σ,

(1) $\to^*_{C(R) \cup E} \cap (\Sigma^* \times \Sigma^*) \subseteq \to^*_R$ *(correctness)*
(2) $\to^*_R \subseteq \to^*_C \circ \to^*_E$, *(completeness)*
for every context-free conjugate C of R

Theorem

Every SRS R has a context-free conjugate C and

- (C, E) *is a* $(\mathrm{CF}, \mathrm{SN} \cap \mathrm{CF}_0^-)$*-decomposition of R*
- $(C(R), E)$ *is a decomposition of R*

Deleting string rewriting

Definition (Deleting string rewriting)

A SRS R over Σ is called deleting
if there is an irreflexive partial ordering $>$ on Σ such that

$$\forall(\ell \to r) \in R \; \exists a \in \ell \; \forall b \in r : a > b$$

Lemma

For a SRS R, the following conditions are equivalent:

(1) *There is a terminating context-free conjugate of R.*

(2) *R is deleting.*

Corollary

Let R be a deleting string rewriting system, then

(1) *R has a $(\mathsf{SN} \cap \mathsf{CF}, \mathsf{SN} \cap \mathsf{CF}_0^-)$-decomposition, and*

(2) *[HofWal04] R preserves regularity, R^- preserves context-freeness.*

Deleting string rewriting

Definition (Deleting string rewriting)

A SRS R over Σ is called deleting
if there is an irreflexive partial ordering $>$ on Σ such that

$$\forall(\ell \to r) \in R \; \exists a \in \ell \; \forall b \in r : a > b$$

Lemma

For a SRS R, the following conditions are equivalent:

(1) *There is a terminating context-free conjugate of R.*

(2) *R is deleting.*

Corollary

Let R be a deleting string rewriting system, then

(1) *R has a $(\mathrm{SN} \cap \mathrm{CF}, \mathrm{SN} \cap \mathrm{CF}_0^-)$-decomposition, and*

(2) *[HofWal04] R preserves regularity, R^- preserves context-freeness.*

Example

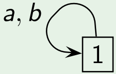
$$R = \{ba \to cb,\ bd \to d,\ cd \to de,\ d \to \epsilon\}$$

is deleting with respect to the ordering $a > b > c > d > e$.
A terminating context-free conjugate of R is

$$C = \{a \to \overleftarrow{b}\, cb,\ b \to d\, \overrightarrow{d},\ c \to de\, \overrightarrow{d},\ d \to \epsilon\}$$

We construct an automaton A with $\mathcal{L}(A) = R^*(L)$ where $L = \{a, b\}^*$.

Naive algorithm: closure under R.



Example

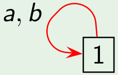
$$R = \{ba \to cb,\ bd \to d,\ cd \to de,\ d \to \epsilon\}$$

is deleting with respect to the ordering $a > b > c > d > e$.
A terminating context-free conjugate of R is

$$C = \{a \to \overleftarrow{b}\, cb,\ b \to d\,\overrightarrow{d},\ c \to de\,\overrightarrow{d},\ d \to \epsilon\}$$

We construct an automaton A with $\mathcal{L}(A) = R^*(L)$ where $L = \{a, b\}^*$.

Naive algorithm: closure under R.



Example

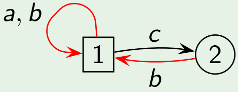
$$R = \{ba \to cb,\ bd \to d,\ cd \to de,\ d \to \epsilon\}$$

is deleting with respect to the ordering $a > b > c > d > e$.
A terminating context-free conjugate of R is

$$C = \{a \to \overleftarrow{b}\,cb,\ b \to d\,\overrightarrow{d},\ c \to de\,\overrightarrow{d},\ d \to \epsilon\}$$

We construct an automaton A with $\mathcal{L}(A) = R^*(L)$ where $L = \{a, b\}^*$.

Naive algorithm: closure under R.



Example

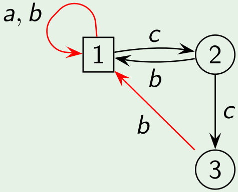
$$R = \{ba \to cb,\ bd \to d,\ cd \to de,\ d \to \epsilon\}$$

is deleting with respect to the ordering $a > b > c > d > e$.
A terminating context-free conjugate of R is

$$C = \{a \to \overleftarrow{b}\,cb,\ b \to d\,\overrightarrow{d},\ c \to de\,\overrightarrow{d},\ d \to \epsilon\}$$

We construct an automaton A with $\mathcal{L}(A) = R^*(L)$ where $L = \{a, b\}^*$.

Naive algorithm: closure under R.



Example

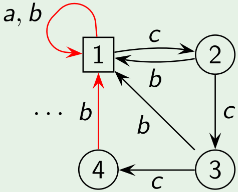
$$R = \{ba \to cb,\ bd \to d,\ cd \to de,\ d \to \epsilon\}$$

is deleting with respect to the ordering $a > b > c > d > e$.
A terminating context-free conjugate of R is

$$C = \{a \to \overleftarrow{b}\, cb,\ b \to d\, \overrightarrow{d},\ c \to de\, \overrightarrow{d},\ d \to \epsilon\}$$

We construct an automaton A with $\mathcal{L}(A) = R^*(L)$ where $L = \{a, b\}^*$.

Naive algorithm: closure under R.



$\Longrightarrow$ the algorithm does not terminate!

Example (…continued)

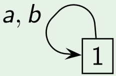
$$R = \{ba \to cb,\ bd \to d,\ cd \to de,\ d \to \epsilon\}$$

$$C = \{a \to \overleftarrow{b}\, cb,\ b \to d\, \overrightarrow{d},\ c \to de\, \overrightarrow{d},\ d \to \epsilon\}$$

$$E = \{a\overleftarrow{a} \to \epsilon,\ \overrightarrow{a}\, a \to \epsilon,\ b\overleftarrow{b} \to \epsilon,\ \overrightarrow{b}\, b \to \epsilon,\ \ldots,\ d\overleftarrow{d} \to \epsilon,\ \overrightarrow{d}\, d \to \epsilon\}$$

We construct an automaton A with $\mathcal{L}(A) = R^*(L)$ where $L = \{a, b\}^*$.

Off-line construction: use $\to_R^* = (\to_C^* \circ \to_E^*) \cap (\Sigma^* \times \Sigma^*)$



1. closure under C
2. closure under E
3. restriction to Σ

Example (...continued)

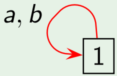
$$R = \{ba \to cb,\ bd \to d,\ cd \to de,\ d \to \epsilon\}$$

$$C = \{a \to \overleftarrow{b}\, cb,\ b \to d\,\overrightarrow{d},\ c \to de\,\overrightarrow{d},\ d \to \epsilon\}$$

$$E = \{a\overleftarrow{a} \to \epsilon,\ \overrightarrow{a}\,a \to \epsilon,\ b\overleftarrow{b} \to \epsilon,\ \overrightarrow{b}\,b \to \epsilon,\ \ldots,\ d\overleftarrow{d} \to \epsilon,\ \overrightarrow{d}\,d \to \epsilon\}$$

We construct an automaton A with $\mathcal{L}(A) = R^*(L)$ where $L = \{a, b\}^*$.

Off-line construction: use $\to_R^* = (\to_C^* \circ \to_E^*) \cap (\Sigma^* \times \Sigma^*)$



1. closure under C
2. closure under E
3. restriction to Σ

Example (…continued)

$$R = \{ba \to cb,\ bd \to d,\ cd \to de,\ d \to \epsilon\}$$

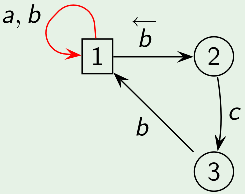
$$C = \{a \to \overleftarrow{b}\,cb,\ b \to d\,\overrightarrow{d},\ c \to de\,\overrightarrow{d},\ d \to \epsilon\}$$

$$E = \{a\overleftarrow{a} \to \epsilon,\ \overrightarrow{a}a \to \epsilon,\ b\overleftarrow{b} \to \epsilon,\ \overrightarrow{b}b \to \epsilon,\ \ldots,\ d\overleftarrow{d} \to \epsilon,\ \overrightarrow{d}d \to \epsilon\}$$

We construct an automaton A with $\mathcal{L}(A) = R^*(L)$ where $L = \{a, b\}^*$.

Off-line construction: use $\to_R^* = (\to_C^* \circ \to_E^*) \cap (\Sigma^* \times \Sigma^*)$

1. closure under C
2. closure under E
3. restriction to Σ



Example (...continued)

$$R = \{ba \to cb,\ bd \to d,\ cd \to de,\ d \to \epsilon\}$$

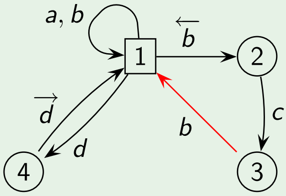
$$C = \{a \to \overleftarrow{b}\, cb,\ b \to d\,\overrightarrow{d},\ c \to de\,\overrightarrow{d},\ d \to \epsilon\}$$

$$E = \{a\overleftarrow{a} \to \epsilon,\ \overrightarrow{a}\, a \to \epsilon,\ b\overleftarrow{b} \to \epsilon,\ \overrightarrow{b}\, b \to \epsilon,\ \dots,\ d\overleftarrow{d} \to \epsilon,\ \overrightarrow{d}\, d \to \epsilon\}$$

We construct an automaton A with $\mathcal{L}(A) = R^*(L)$ where $L = \{a, b\}^*$.

Off-line construction: use $\to_R^* = (\to_C^* \circ \to_E^*) \cap (\Sigma^* \times \Sigma^*)$

1. closure under C
2. closure under E
3. restriction to Σ



Example (...continued)

$$R = \{ba \to cb,\ bd \to d,\ cd \to de,\ d \to \epsilon\}$$

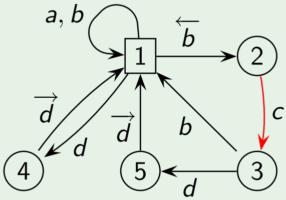
$$C = \{a \to \overleftarrow{b}\,cb,\ b \to d\,\overrightarrow{d},\ c \to de\,\overrightarrow{d},\ d \to \epsilon\}$$

$$E = \{a\overleftarrow{a} \to \epsilon,\ \overrightarrow{a}\,a \to \epsilon,\ b\overleftarrow{b} \to \epsilon,\ \overrightarrow{b}\,b \to \epsilon,\ \dots,\ d\overleftarrow{d} \to \epsilon,\ \overrightarrow{d}\,d \to \epsilon\}$$

We construct an automaton A with $\mathcal{L}(A) = R^*(L)$ where $L = \{a, b\}^*$.

Off-line construction: use $\to_R^* = (\to_C^* \circ \to_E^*) \cap (\Sigma^* \times \Sigma^*)$

1. closure under C
2. closure under E
3. restriction to Σ



Example (. . . continued)

$$R = \{ba \to cb,\ bd \to d,\ cd \to de,\ d \to \epsilon\}$$

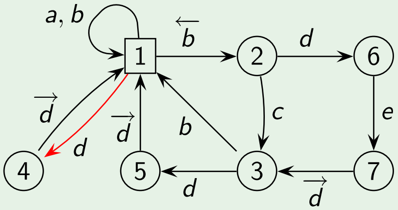
$$C = \{a \to \overleftarrow{b}\,cb,\ b \to d\,\overrightarrow{d},\ c \to de\,\overrightarrow{d},\ d \to \epsilon\}$$

$$E = \{a\overleftarrow{a} \to \epsilon,\ \overrightarrow{a}a \to \epsilon,\ b\overleftarrow{b} \to \epsilon,\ \overrightarrow{b}b \to \epsilon,\ \dots,\ d\overleftarrow{d} \to \epsilon,\ \overrightarrow{d}d \to \epsilon\}$$

We construct an automaton A with $\mathcal{L}(A) = R^*(L)$ where $L = \{a, b\}^*$.

Off-line construction: use $\to_R^* = (\to_C^* \circ \to_E^*) \cap (\Sigma^* \times \Sigma^*)$

1. closure under C
2. closure under E
3. restriction to Σ



Example (…continued)

$$R = \{ba \to cb,\ bd \to d,\ cd \to de,\ d \to \epsilon\}$$

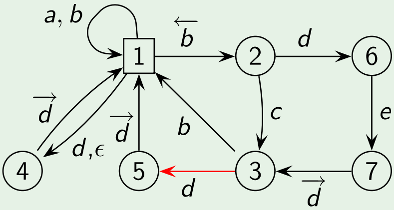
$$C = \{a \to \overleftarrow{b}\, cb,\ b \to d\,\overrightarrow{d},\ c \to de\,\overrightarrow{d},\ d \to \epsilon\}$$

$$E = \{a\overleftarrow{a} \to \epsilon,\ \overrightarrow{a}\,a \to \epsilon,\ b\overleftarrow{b} \to \epsilon,\ \overrightarrow{b}\,b \to \epsilon,\ \ldots,\ d\overleftarrow{d} \to \epsilon,\ \overrightarrow{d}\,d \to \epsilon\}$$

We construct an automaton A with $\mathcal{L}(A) = R^*(L)$ where $L = \{a, b\}^*$.

Off-line construction: use $\to_R^* = (\to_C^* \circ \to_E^*) \cap (\Sigma^* \times \Sigma^*)$

1. closure under C
2. closure under E
3. restriction to Σ



Example (...continued)

$$R = \{ba \to cb,\ bd \to d,\ cd \to de,\ d \to \epsilon\}$$

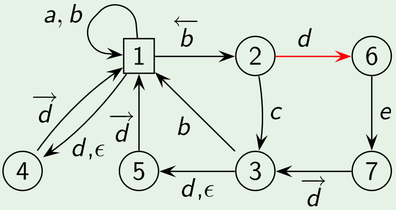
$$C = \{a \to \overleftarrow{b}\,cb,\ b \to d\,\overrightarrow{d},\ c \to de\,\overrightarrow{d},\ d \to \epsilon\}$$

$$E = \{a\overleftarrow{a} \to \epsilon,\ \overrightarrow{a}a \to \epsilon,\ b\overleftarrow{b} \to \epsilon,\ \overrightarrow{b}b \to \epsilon,\ \ldots,\ d\overleftarrow{d} \to \epsilon,\ \overrightarrow{d}d \to \epsilon\}$$

We construct an automaton A with $\mathcal{L}(A) = R^*(L)$ where $L = \{a, b\}^*$.

Off-line construction: use $\to_R^* = (\to_C^* \circ \to_E^*) \cap (\Sigma^* \times \Sigma^*)$

1. closure under C
2. closure under E
3. restriction to Σ



Example (…continued)

$$R = \{ba \to cb,\ bd \to d,\ cd \to de,\ d \to \epsilon\}$$

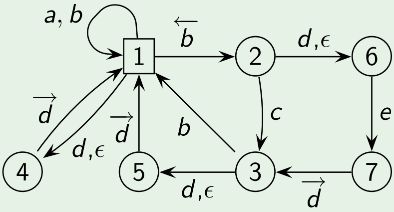
$$C = \{a \to \overleftarrow{b}\,cb,\ b \to d\,\overrightarrow{d},\ c \to de\,\overrightarrow{d},\ d \to \epsilon\}$$

$$E = \{a\overleftarrow{a} \to \epsilon,\ \overrightarrow{a}\,a \to \epsilon,\ b\overleftarrow{b} \to \epsilon,\ \overrightarrow{b}\,b \to \epsilon,\ \ldots,\ d\overleftarrow{d} \to \epsilon,\ \overrightarrow{d}\,d \to \epsilon\}$$

We construct an automaton A with $\mathcal{L}(A) = R^*(L)$ where $L = \{a, b\}^*$.

Off-line construction: use $\to_R^* = (\to_C^* \circ \to_E^*) \cap (\Sigma^* \times \Sigma^*)$

1. closure under C
2. closure under E
3. restriction to Σ



Example (...continued)

$$R = \{ba \to cb,\ bd \to d,\ cd \to de,\ d \to \epsilon\}$$

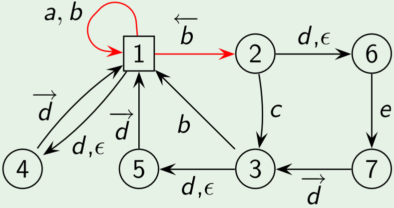
$$C = \{a \to \overleftarrow{b}\,cb,\ b \to d\,\overrightarrow{d},\ c \to de\,\overrightarrow{d},\ d \to \epsilon\}$$

$$E = \{a\overleftarrow{a} \to \epsilon,\ \overrightarrow{a}\,a \to \epsilon,\ b\overleftarrow{b} \to \epsilon,\ \overrightarrow{b}\,b \to \epsilon,\ \ldots,\ d\overleftarrow{d} \to \epsilon,\ \overrightarrow{d}\,d \to \epsilon\}$$

We construct an automaton A with $\mathcal{L}(A) = R^*(L)$ where $L = \{a, b\}^*$.

Off-line construction: use $\to_R^* = (\to_C^* \circ \to_E^*) \cap (\Sigma^* \times \Sigma^*)$

1. closure under C
2. closure under E
3. restriction to Σ



Example (…continued)

$$R = \{ba \to cb,\ bd \to d,\ cd \to de,\ d \to \epsilon\}$$

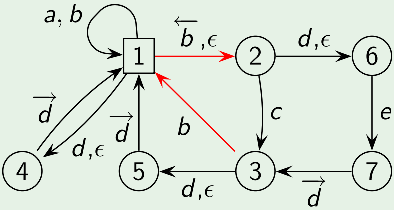
$$C = \{a \to \overleftarrow{b}\,cb,\ b \to d\,\overrightarrow{d},\ c \to de\,\overrightarrow{d},\ d \to \epsilon\}$$

$$E = \{a\overleftarrow{a} \to \epsilon,\ \overrightarrow{a}\,a \to \epsilon,\ b\overleftarrow{b} \to \epsilon,\ \overrightarrow{b}\,b \to \epsilon,\ \ldots,\ d\overleftarrow{d} \to \epsilon,\ \overrightarrow{d}\,d \to \epsilon\}$$

We construct an automaton A with $\mathcal{L}(A) = R^*(L)$ where $L = \{a,b\}^*$.

Off-line construction: use $\to_R^* = (\to_C^* \circ \to_E^*) \cap (\Sigma^* \times \Sigma^*)$

1. closure under C
2. closure under E
3. restriction to Σ



Example (…continued)

$$R = \{ba \to cb,\ bd \to d,\ cd \to de,\ d \to \epsilon\}$$

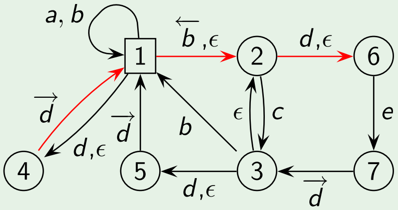
$$C = \{a \to \overleftarrow{b}\,cb,\ b \to d\,\overrightarrow{d},\ c \to de\,\overrightarrow{d},\ d \to \epsilon\}$$

$$E = \{a\overleftarrow{a} \to \epsilon,\ \overrightarrow{a}a \to \epsilon,\ b\overleftarrow{b} \to \epsilon,\ \overrightarrow{b}b \to \epsilon,\ \ldots,\ d\overleftarrow{d} \to \epsilon,\ \overrightarrow{d}d \to \epsilon\}$$

We construct an automaton A with $\mathcal{L}(A) = R^*(L)$ where $L = \{a, b\}^*$.

Off-line construction: use $\to_R^* = (\to_C^* \circ \to_E^*) \cap (\Sigma^* \times \Sigma^*)$

1. closure under C
2. closure under E
3. restriction to Σ



Example (...continued)

$$R = \{ba \to cb,\ bd \to d,\ cd \to de,\ d \to \epsilon\}$$

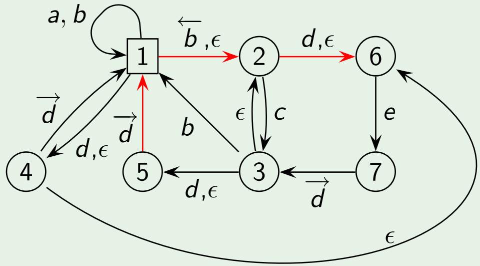
$$C = \{a \to \overleftarrow{b}\,cb,\ b \to d\,\overrightarrow{d},\ c \to de\,\overrightarrow{d},\ d \to \epsilon\}$$

$$E = \{a\overleftarrow{a} \to \epsilon,\ \overrightarrow{a}a \to \epsilon,\ b\overleftarrow{b} \to \epsilon,\ \overrightarrow{b}b \to \epsilon,\ \ldots,\ d\overleftarrow{d} \to \epsilon,\ \overrightarrow{d}d \to \epsilon\}$$

We construct an automaton A with $\mathcal{L}(A) = R^*(L)$ where $L = \{a, b\}^*$.

Off-line construction: use $\to_R^* = (\to_C^* \circ \to_E^*) \cap (\Sigma^* \times \Sigma^*)$

1. closure under C
2. closure under E
3. restriction to Σ



Example (. . . continued)

$$R = \{ba \to cb,\ bd \to d,\ cd \to de,\ d \to \epsilon\}$$

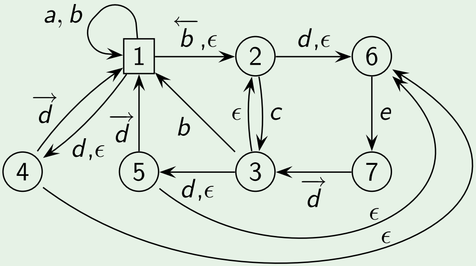
$$C = \{a \to \overleftarrow{b}\, cb,\ b \to d\, \overrightarrow{d},\ c \to de\, \overrightarrow{d},\ d \to \epsilon\}$$

$$E = \{a\overleftarrow{a} \to \epsilon,\ \overrightarrow{a}\, a \to \epsilon,\ b\overleftarrow{b} \to \epsilon,\ \overrightarrow{b}\, b \to \epsilon,\ \ldots,\ d\overleftarrow{d} \to \epsilon,\ \overrightarrow{d}\, d \to \epsilon\}$$

We construct an automaton A with $\mathcal{L}(A) = R^*(L)$ where $L = \{a, b\}^*$.

Off-line construction: use $\to_R^* = (\to_C^* \circ \to_E^*) \cap (\Sigma^* \times \Sigma^*)$

1. closure under C
2. closure under E
3. restriction to Σ



Example (...continued)

$$R = \{ba \to cb,\ bd \to d,\ cd \to de,\ d \to \epsilon\}$$

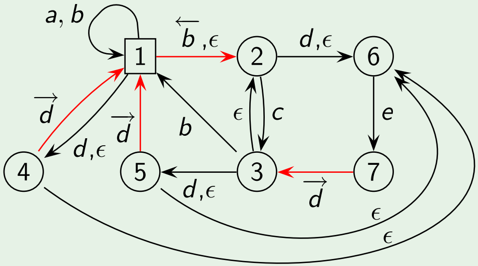
$$C = \{a \to \overleftarrow{b}\,cb,\ b \to d\,\overrightarrow{d},\ c \to de\,\overrightarrow{d},\ d \to \epsilon\}$$

$$E = \{a\overleftarrow{a} \to \epsilon,\ \overrightarrow{a}\,a \to \epsilon,\ b\overleftarrow{b} \to \epsilon,\ \overrightarrow{b}\,b \to \epsilon,\ \dots,\ d\overleftarrow{d} \to \epsilon,\ \overrightarrow{d}\,d \to \epsilon\}$$

We construct an automaton A with $\mathcal{L}(A) = R^*(L)$ where $L = \{a, b\}^*$.

Off-line construction: use $\to_R^* = (\to_C^* \circ \to_E^*) \cap (\Sigma^* \times \Sigma^*)$

1. closure under C
2. closure under E
3. restriction to Σ



Example (...continued)

$$R = \{ba \to cb,\ bd \to d,\ cd \to de,\ d \to \epsilon\}$$

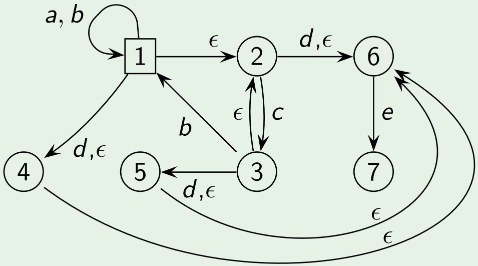
$$C = \{a \to \overleftarrow{b}\, cb,\ b \to d\,\overrightarrow{d},\ c \to de\,\overrightarrow{d},\ d \to \epsilon\}$$

$$E = \{a\overleftarrow{a} \to \epsilon,\ \overrightarrow{a}\,a \to \epsilon,\ b\overleftarrow{b} \to \epsilon,\ \overrightarrow{b}\,b \to \epsilon,\ \dots,\ d\overleftarrow{d} \to \epsilon,\ \overrightarrow{d}\,d \to \epsilon\}$$

We construct an automaton A with $\mathcal{L}(A) = R^*(L)$ where $L = \{a, b\}^*$.

Off-line construction: use $\to_R^* = (\to_C^* \circ \to_E^*) \cap (\Sigma^* \times \Sigma^*)$

1. closure under C
2. closure under E
3. restriction to Σ



Example (...continued)

$$R = \{ba \to cb,\ bd \to d,\ cd \to de,\ d \to \epsilon\}$$

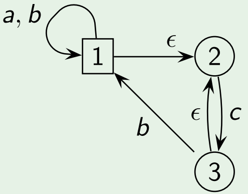
$$C = \{a \to \overleftarrow{b}\, cb,\ b \to d\,\overrightarrow{d},\ c \to de\,\overrightarrow{d},\ d \to \epsilon\}$$

$$E = \{a\overleftarrow{a} \to \epsilon,\ \overrightarrow{a}\,a \to \epsilon,\ b\overleftarrow{b} \to \epsilon,\ \overrightarrow{b}\,b \to \epsilon,\ \ldots,\ d\overleftarrow{d} \to \epsilon,\ \overrightarrow{d}\,d \to \epsilon\}$$

We construct an automaton A with $\mathcal{L}(A) = R^*(L)$ where $L = \{a, b\}^*$.

Off-line construction: use $\to_R^* = (\to_C^* \circ \to_E^*) \cap (\Sigma^* \times \Sigma^*)$

1. closure under C
2. closure under E
3. restriction to Σ



$$\Longrightarrow R^*(L) = (a + c^* b)^*$$

Match-bounded string rewriting

Following [GesHofWal04], we annotate letters by numbers.

- Extended alphabet: $\Gamma = \Sigma \times \mathbb{N}$ (we write a_n for (a, n) in Γ)

Define base : $\Gamma \to \Sigma$, height : $\Gamma \to \mathbb{N}$, $\text{lift}_n : \Sigma \to \Gamma$ for $n \in \mathbb{N}$ by

$$\text{base}(a_n) = a,\ \text{height}(a_n) = n,\ \text{lift}_n(a) = a_n.$$

Definition (match(R))

For a SRS R over Σ where $\epsilon \notin \text{lhs}(R)$ define a SRS over Γ:

$$\text{match}(R) = \{\ell' \to \text{lift}_{m+1}(r) \mid (\ell \to r) \in R,\ \text{base}(\ell') = \ell,\\ m = \min \text{height}(\ell')\}$$

The SRS match(R) simulates R-rewriting: $\to_R^* = \text{lift}_0 \circ \to_{\text{match}(R)}^* \circ \text{base}$

Example (match($\{aa \to aba\}$))

$a_0 a_0 \to a_1 b_1 a_1,\ a_0 a_1 \to a_1 b_1 a_1,\ a_2 a_1 \to a_2 b_2 a_2,\ a_4 a_8 \to a_5 b_5 a_5,\ \ldots$

Definition (Match-boundedness)

The system R is called match-bounded by $h \in \mathbb{N}$ if

$$\to^*_{\mathrm{match}(R)}(\mathrm{lift}_0(\Sigma^*)) \subseteq (\Sigma \times \{0, \ldots, h\})^*$$

Theorem

Every match-bounded SRS is terminating.

For a system S over $\Sigma \times \mathbb{N}$ let $S_h = S|_{\Sigma \times \{0,\ldots,h\}}$.

If R is match-bounded by h then $\to^*_R = \mathrm{lift}_0 \circ \to^*_{\mathrm{match}_h(R)} \circ \mathrm{base}$

Remark

Each system $\mathrm{match}_h(R)$ *is deleting w.r.t.* $a_m > b_n$ *if* $m < n$. *Hence we could apply the decomposition for deleting systems to* $\mathrm{match}_h(R)$, *but . . .*

Definition (Match-boundedness)

The system R is called match-bounded by $h \in \mathbb{N}$ if

$$\to^*_{\mathrm{match}(R)}(\mathrm{lift}_0(\Sigma^*)) \subseteq (\Sigma \times \{0, \ldots, h\})^*$$

Theorem

Every match-bounded SRS is terminating.

For a system S over $\Sigma \times \mathbb{N}$ let $S_h = S|_{\Sigma \times \{0,\ldots,h\}}$.

If R is match-bounded by h then $\to^*_R = \mathrm{lift}_0 \circ \to^*_{\mathrm{match}_h(R)} \circ \mathrm{base}$

Remark

Each system $\mathrm{match}_h(R)$ *is deleting w.r.t.* $a_m > b_n$ *if* $m < n$. *Hence we could apply the decomposition for deleting systems to* $\mathrm{match}_h(R)$, *but ...*

Improved decomposition for match(R)

Giving up uniqueness of the inverses to improve the decomposition:

$$E' = \{ \overrightarrow{a_i}\, a_j \to \epsilon,\ a_j \overleftarrow{a_i} \to \epsilon \mid a \in \Sigma,\ j \geq i \geq 0 \}$$

$$C' = \{ a_i \to \mathrm{lift}_i(\overleftarrow{x})\, \mathrm{lift}_{i+1}(r)\, \mathrm{lift}_i(\overrightarrow{y}) \mid (xay \to r) \in R,\ a \in \Sigma,\ i \geq 0 \}$$

Example

Take $R = \{aa \to aba\}$, and consider decompositions of $\mathrm{match}_2(R)$.

$$C_2' = \{\ a_0 \to \overleftarrow{a_0} a_1 b_1 a_1,\ a_0 \to a_1 b_1 a_1 \overrightarrow{a_0},\ a_1 \to \overleftarrow{a_1} a_2 b_2 a_2,\ a_1 \to a_2 b_2 a_2 \overrightarrow{a_1}\ \}$$

$$C_2 = C_2' \uplus \{\ a_0 \to \overleftarrow{a_1} a_1 b_1 a_1,\ a_0 \to \overleftarrow{a_2} a_1 b_1 a_1,\ a_0 \to a_1 b_1 a_1 \overrightarrow{a_1},\ a_0 \to a_1 b_1 a_1 \overrightarrow{a_2},\ a_1 \to \overleftarrow{a_2} a_2 b_2 a_2,\ a_1 \to a_2 b_2 a_2 \overrightarrow{a_2}\ \}$$

C_2' contains 4 rules, and $E_2' = \{\overrightarrow{a_0} a_0 \to \epsilon, \overrightarrow{a_0} a_1 \to \epsilon, \ldots\}$ with 24 rules. In contrast, $C_2' \subset C_2$ with $|C_2| = 10$, while $E_2 \subset E_2'$ and $|E_2| = 12$.

Theorem (match(R) decomposition)

(C', E') *is a* $(\mathrm{SN} \cap \mathrm{CF}, \mathrm{SN} \cap \mathrm{CF}_0^-)$*-decomposition of* $\mathrm{match}(R)$.
(C'_h, E'_h) *is a* $(\mathrm{SN} \cap \mathrm{CF}, \mathrm{SN} \cap \mathrm{CF}_0^-)$*-decomposition of* $\mathrm{match}_h(R)$.

Corollary

Every match-bounded SRS has a $(\mathrm{SN} \cap \mathrm{CF}, \mathrm{SN} \cap \mathrm{CF}^-)$*-decomposition.*

The improved decomposition yields a drastic reduction from C_c to C'_c:

$$|C_h| \leq |R| \cdot m \cdot (h+1)^m \qquad |E_h| = |\Sigma| \cdot O(h)$$

$$|C'_h| \leq |R| \cdot m \cdot h \qquad |E'_h| = |\Sigma| \cdot O(h^2)$$

Less rules in C imply smaller automata. E adds only ϵ-transitions.

Example

$$R = \{caac \to aaa,\ b \to aca,\ aba \to bb\} \text{ (RFC-match-bound 12)}$$

$$|C_{12}| = 64054 \qquad |C'_{12}| = 286 \qquad |E_{12}| = 78 \qquad |E'_{12}| = 546$$

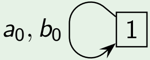
$$|A| > 10^{45} \qquad |A'| < 10^{15}$$

where $|A|$, $|A'|$ are the automata sizes using the off-line construction.

Example (On-line construction for match-bound search)

$$R = \{aa \to aba\} \text{ over } \Sigma = \{a, b\}$$

On-line construction: for every R-redex we choose a conjugate $a \to r$, add a fresh path labeled with r, followed by closure under E.

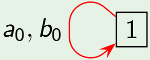


- Automaton for $\text{lift}_0(L)$
- $1 \xrightarrow{a_0} 1 \xrightarrow{a_0} 1$ is a match(R)-redex $\Longrightarrow$ conjugate $a_0 \to a_1 b_1 a_1 \overrightarrow{a_0}$
- $4 \xrightarrow{\overrightarrow{a_0}} 1 \xrightarrow{a_0} 1$ is an E'-redex, we add a transition $4 \xrightarrow{\epsilon} 1$
- $4 \xrightarrow{\overrightarrow{a_0}} 1 \xrightarrow{a_1} 2$ is an E'-redex, we add a transition $4 \xrightarrow{\epsilon} 2$
- $3 \xrightarrow{a_1} 4 \xrightarrow{\epsilon} 1 \xrightarrow{a_1} 2$ is a match(R)-redex $\Longrightarrow$ conjugate $a_1 \to a_2 b_2 a_2 \overrightarrow{a_1}$
- $7 \xrightarrow{\overrightarrow{a_1}} 4 \xrightarrow{\epsilon} 1 \xrightarrow{a_1} 2$ is an E'-redex, we add a transition $7 \xrightarrow{\epsilon} 2$
- Now the automaton is compatible with match(R).
  Hence R is match-bounded by 2 and therefore terminating.

Example (On-line construction for match-bound search)

$$R = \{aa \to aba\} \text{ over } \Sigma = \{a, b\}$$

On-line construction: for every R-redex we choose a conjugate $a \to r$, add a fresh path labeled with r, followed by closure under E.

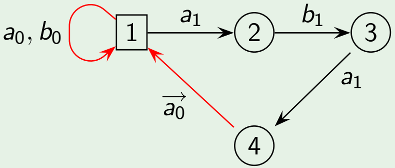


- Automaton for $\text{lift}_0(L)$
- $1 \xrightarrow{a_0} 1 \xrightarrow{a_0} 1$ is a match(R)-redex $\Longrightarrow$ conjugate $a_0 \to a_1 b_1 a_1 \overrightarrow{a_0}$
- $4 \xrightarrow{\overrightarrow{a_0}} 1 \xrightarrow{a_0} 1$ is an E'-redex, we add a transition $4 \xrightarrow{\epsilon} 1$
- $4 \xrightarrow{\overrightarrow{a_0}} 1 \xrightarrow{a_1} 2$ is an E'-redex, we add a transition $4 \xrightarrow{\epsilon} 2$
- $3 \xrightarrow{a_1} 4 \xrightarrow{\epsilon} 1 \xrightarrow{a_1} 2$ is a match(R)-redex $\Longrightarrow$ conjugate $a_1 \to a_2 b_2 a_2 \overrightarrow{a_1}$
- $7 \xrightarrow{\overrightarrow{a_1}} 4 \xrightarrow{\epsilon} 1 \xrightarrow{a_1} 2$ is an E'-redex, we add a transition $7 \xrightarrow{\epsilon} 2$
- Now the automaton is compatible with match(R).
  Hence R is match-bounded by 2 and therefore terminating.

Example (On-line construction for match-bound search)

$$R = \{aa \to aba\} \text{ over } \Sigma = \{a, b\}$$

On-line construction: for every R-redex we choose a conjugate $a \to r$, add a fresh path labeled with r, followed by closure under E.

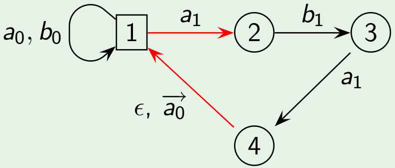


- Automaton for $\text{lift}_0(L)$
- $1 \xrightarrow{a_0} 1 \xrightarrow{a_0} 1$ is a match(R)-redex $\Longrightarrow$ conjugate $a_0 \to a_1 b_1 a_1 \overrightarrow{a_0}$
- $4 \xrightarrow{\overrightarrow{a_0}} 1 \xrightarrow{a_0} 1$ is an E'-redex, we add a transition $4 \xrightarrow{\epsilon} 1$
- $4 \xrightarrow{\overrightarrow{a_0}} 1 \xrightarrow{a_1} 2$ is an E'-redex, we add a transition $4 \xrightarrow{\epsilon} 2$
- $3 \xrightarrow{a_1} 4 \xrightarrow{\epsilon} 1 \xrightarrow{a_1} 2$ is a match(R)-redex $\Longrightarrow$ conjugate $a_1 \to a_2 b_2 a_2 \overrightarrow{a_1}$
- $7 \xrightarrow{\overrightarrow{a_1}} 4 \xrightarrow{\epsilon} 1 \xrightarrow{a_1} 2$ is an E'-redex, we add a transition $7 \xrightarrow{\epsilon} 2$
- Now the automaton is compatible with match(R).
  Hence R is match-bounded by 2 and therefore terminating.

Example (On-line construction for match-bound search)

$$R = \{aa \to aba\} \text{ over } \Sigma = \{a, b\}$$

On-line construction: for every R-redex we choose a conjugate $a \to r$, add a fresh path labeled with r, followed by closure under E.

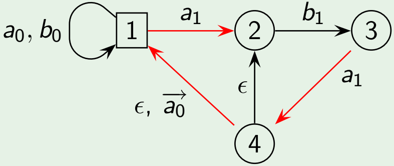


- Automaton for $\text{lift}_0(L)$
- $1 \xrightarrow{a_0} 1 \xrightarrow{a_0} 1$ is a match(R)-redex $\Longrightarrow$ conjugate $a_0 \to a_1 b_1 a_1 \overrightarrow{a_0}$
- $4 \xrightarrow{\overrightarrow{a_0}} 1 \xrightarrow{a_0} 1$ is an E'-redex, we add a transition $4 \xrightarrow{\epsilon} 1$
- $4 \xrightarrow{\overrightarrow{a_0}} 1 \xrightarrow{a_1} 2$ is an E'-redex, we add a transition $4 \xrightarrow{\epsilon} 2$
- $3 \xrightarrow{a_1} 4 \xrightarrow{\epsilon} 1 \xrightarrow{a_1} 2$ is a match(R)-redex $\Longrightarrow$ conjugate $a_1 \to a_2 b_2 a_2 \overrightarrow{a_1}$
- $7 \xrightarrow{\overrightarrow{a_1}} 4 \xrightarrow{\epsilon} 1 \xrightarrow{a_1} 2$ is an E'-redex, we add a transition $7 \xrightarrow{\epsilon} 2$
- Now the automaton is compatible with match(R).
  Hence R is match-bounded by 2 and therefore terminating.

Example (On-line construction for match-bound search)

$$R = \{aa \to aba\} \text{ over } \Sigma = \{a, b\}$$

On-line construction: for every R-redex we choose a conjugate $a \to r$, add a fresh path labeled with r, followed by closure under E.

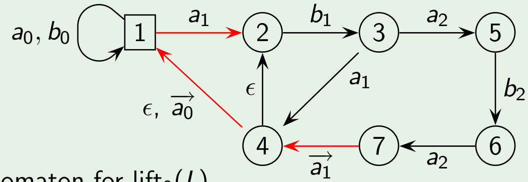


- Automaton for $\mathrm{lift}_0(L)$
- $1 \xrightarrow{a_0} 1 \xrightarrow{a_0} 1$ is a match(R)-redex $\Longrightarrow$ conjugate $a_0 \to a_1 b_1 a_1 \overrightarrow{a_0}$
- $4 \xrightarrow{\overrightarrow{a_0}} 1 \xrightarrow{a_0} 1$ is an E'-redex, we add a transition $4 \xrightarrow{\epsilon} 1$
- $4 \xrightarrow{\overrightarrow{a_0}} 1 \xrightarrow{a_1} 2$ is an E'-redex, we add a transition $4 \xrightarrow{\epsilon} 2$
- $3 \xrightarrow{a_1} 4 \xrightarrow{\epsilon} 1 \xrightarrow{a_1} 2$ is a match(R)-redex $\Longrightarrow$ conjugate $a_1 \to a_2 b_2 a_2 \overrightarrow{a_1}$
- $7 \xrightarrow{\overrightarrow{a_1}} 4 \xrightarrow{\epsilon} 1 \xrightarrow{a_1} 2$ is an E'-redex, we add a transition $7 \xrightarrow{\epsilon} 2$
- Now the automaton is compatible with match(R).
  Hence R is match-bounded by 2 and therefore terminating.

Example (On-line construction for match-bound search)

$$R = \{aa \to aba\} \text{ over } \Sigma = \{a, b\}$$

On-line construction: for every R-redex we choose a conjugate $a \to r$, add a fresh path labeled with r, followed by closure under E.

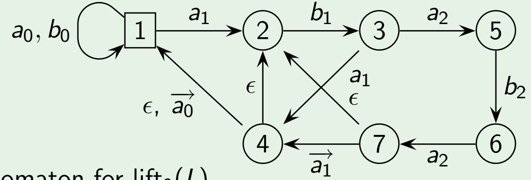


- Automaton for $\mathrm{lift}_0(L)$
- $1 \xrightarrow{a_0} 1 \xrightarrow{a_0} 1$ is a match(R)-redex $\Longrightarrow$ conjugate $a_0 \to a_1 b_1 a_1 \overrightarrow{a_0}$
- $4 \xrightarrow{\overrightarrow{a_0}} 1 \xrightarrow{a_0} 1$ is an E'-redex, we add a transition $4 \xrightarrow{\epsilon} 1$
- $4 \xrightarrow{\overrightarrow{a_0}} 1 \xrightarrow{a_1} 2$ is an E'-redex, we add a transition $4 \xrightarrow{\epsilon} 2$
- $3 \xrightarrow{a_1} 4 \xrightarrow{\epsilon} 1 \xrightarrow{a_1} 2$ is a match(R)-redex $\Longrightarrow$ conjugate $a_1 \to a_2 b_2 a_2 \overrightarrow{a_1}$
- $7 \xrightarrow{\overrightarrow{a_1}} 4 \xrightarrow{\epsilon} 1 \xrightarrow{a_1} 2$ is an E'-redex, we add a transition $7 \xrightarrow{\epsilon} 2$
- Now the automaton is compatible with match(R).
  Hence R is match-bounded by 2 and therefore terminating.

Example (On-line construction for match-bound search)

$$R = \{aa \to aba\} \text{ over } \Sigma = \{a, b\}$$

On-line construction: for every R-redex we choose a conjugate $a \to r$, add a fresh path labeled with r, followed by closure under E.



- Automaton for $\text{lift}_0(L)$
- $1 \xrightarrow{a_0} 1 \xrightarrow{a_0} 1$ is a match(R)-redex $\Longrightarrow$ conjugate $a_0 \to a_1 b_1 a_1 \overrightarrow{a_0}$
- $4 \xrightarrow{\overrightarrow{a_0}} 1 \xrightarrow{a_0} 1$ is an E'-redex, we add a transition $4 \xrightarrow{\epsilon} 1$
- $4 \xrightarrow{\overrightarrow{a_0}} 1 \xrightarrow{a_1} 2$ is an E'-redex, we add a transition $4 \xrightarrow{\epsilon} 2$
- $3 \xrightarrow{a_1} 4 \xrightarrow{\epsilon} 1 \xrightarrow{a_1} 2$ is a match(R)-redex $\Longrightarrow$ conjugate $a_1 \to a_2 b_2 a_2 \overrightarrow{a_1}$
- $7 \xrightarrow{\overrightarrow{a_1}} 4 \xrightarrow{\epsilon} 1 \xrightarrow{a_1} 2$ is an E'-redex, we add a transition $7 \xrightarrow{\epsilon} 2$
- Now the automaton is compatible with match(R).
  Hence R is match-bounded by 2 and therefore terminating.

Example

To conclude, we consider the system

$$R = \{caac \to aaa,\ b \to aca,\ aba \to bb\}.$$

Jambox is the only termination prover that solved this problem in the recent termination competition, see

`http://www.lri.fr/~marche/termination-competition/`,

problem `SRS/secret2006/jambox-1`.

The implementation of our on-line algorithm constructs an exactly compatible automaton with 27.957 states that certifies the RFC-match-bound 12. (time is 1,13 seconds on a Athlon 3200+)

R. V. Book, M. Jantzen, and C. Wrathall.
Monadic Thue systems.
Theoret. Comput. Sci., 19:231–251, 1982.

A. Geser, D. Hofbauer and J. Waldmann.
Match-bounded string rewriting systems.
Appl. Algebra Engrg. Comm. Comput., 15(3-4):149-171, 2004.

A. Geser, D. Hofbauer, J. Waldmann, and H. Zantema.
Finding finite automata that certify termination of string rewriting.
Internat. J. Found. Comput. Sci. 16(3):471–486, 2005.

D. Hofbauer and J. Waldmann.
Deleting string rewriting systems preserve regularity.
Theoret. Comput. Sci., 327(3):301–317, 2004.

J. Sakarovitch.
Eléments de Théorie des Automates.
Vuibert, Paris, 2003.