Uniwersytet Wrocławski

On Recompression for Word Equations

Artur Jeż

Meeting on String Constraints and Applications (MOSCA)
07.05.2019

Definition (Satisfiability of word equations)

Given equation $U = V$, where $U, V \in (\Sigma \cup \mathcal{X})^*$.
Is there a substitution $S : \mathcal{X} \to \Sigma^*$ satisfying the equation?
(Also more general: fintiely many solutions, representation of all, ...)

Definition (Satisfiability of word equations)

Given equation $U = V$, where $U, V \in (\Sigma \cup \mathcal{X})^*$.
Is there a substitution $S : \mathcal{X} \to \Sigma^*$ satisfying the equation?
(Also more general: fintiely many solutions, representation of all, ...)

$$a\,X\,b\,X\,Y\,bbb = X\,abaa\,Y\,b\,Y \qquad S(X) = aa, S(Y) = bb$$

Definition (Satisfiability of word equations)

Given equation $U = V$, where $U, V \in (\Sigma \cup \mathcal{X})^*$.
Is there a substitution $S : \mathcal{X} \to \Sigma^*$ satisfying the equation?
(Also more general: fintiely many solutions, representation of all, . . .)

$$a\,X\,b\,X\,Y\,bbb = X\,abaa\,Y\,b\,Y \qquad S(X) = aa, S(Y) = bb$$
$$a\,aa\,b\,aa\,bb\,bbb = aa\,abaa\,bb\,b\,bb$$

Definition (Satisfiability of word equations)

Given equation $U = V$, where $U, V \in (\Sigma \cup \mathcal{X})^*$.
Is there a substitution $S : \mathcal{X} \to \Sigma^*$ satisfying the equation?
(Also more general: fintiely many solutions, representation of all, . . .)

$$a\,X\,b\,X\,Y\,bbb = X\,abaa\,Y\,b\,Y \qquad S(X) = aa, S(Y) = bb$$
$$a\,aa\,b\,aa\,bb\,bbb = aa\,abaa\,bb\,b\,bb$$

We extend S to a $S : (\Sigma \cup \mathcal{X})^* \to \Sigma^*$; identity on Σ.
$S(U)$ is a solution word.
Lenght-minimal S: minimises $|S(U)|$.
Usually: no $S(X) = \epsilon$, i.e. $S : \mathcal{X} \to \Sigma^+$.

Makanin's algorithm 1977

High complexity [EXPSPACE '98], difficult proof.

Makanin's algorithm 1977

High complexity [EXPSPACE '98], difficult proof.

Compression and word equations

Makanin's algorithm 1977

High complexity [EXPSPACE '98], difficult proof.

Compression and word equations

- Length minimal solution (length N): compressible to poly$(\log N)$. 2NEXPTIME [Plandowski and Rytter, 1998]

Makanin's algorithm 1977

High complexity [EXPSPACE '98], difficult proof.

Compression and word equations

- Length minimal solution (length N): compressible to poly$(\log N)$. 2NEXPTIME [Plandowski and Rytter, 1998]
- The size N of the minimal solution is at most doubly exponential. NEXPTIME [Plandowski 1999]

Makanin's algorithm 1977

High complexity [EXPSPACE '98], difficult proof.

Compression and word equations

- Length minimal solution (length N): compressible to poly$(\log N)$. 2NEXPTIME [Plandowski and Rytter, 1998]
- The size N of the minimal solution is at most doubly exponential. NEXPTIME [Plandowski 1999]
- PSPACE [Plandowski 1999]

Makanin's algorithm 1977

High complexity [EXPSPACE '98], difficult proof.

Compression and word equations

- Length minimal solution (length N): compressible to poly$(\log N)$. 2NEXPTIME [Plandowski and Rytter, 1998]
- The size N of the minimal solution is at most doubly exponential. NEXPTIME [Plandowski 1999]
- PSPACE [Plandowski 1999]
- The same, but simpler. [J. 2013]

Makanin's algorithm 1977

High complexity [EXPSPACE '98], difficult proof.

Compression and word equations

- Length minimal solution (length N): compressible to poly$(\log N)$. 2NEXPTIME [Plandowski and Rytter, 1998]
- The size N of the minimal solution is at most doubly exponential. NEXPTIME [Plandowski 1999]
- PSPACE [Plandowski 1999]
- The same, but simpler. [J. 2013]

Only NP-hard. And believed to be in NP.
Solutions at most exponential?

Simplicity

Simple is good on its own.

Simplicity

Simple is good on its own.

Easier to generalize

- Regular constraints [Diekert, J., Plandowski]
- Involution ($\overline{aw} = \overline{w}\,\overline{a}$) [Diekert, J., Plandowski]
- free groups [Diekert, J., Plandowski]
- generation of all solutions [J.]
  for free groups [Diekert, J., Plandowski]
- partial commutation [Diekert, J., Kufleitner]
- all solutions are EDT0L language [Ciobanu, Diekert, Elder]
- nondeterministic linear space = context sensitive language [J.]
- twisted word equations (permutation of letters) [Diekert, Elder]
- linear time for one variable [J.]
- context unification (terms) [J.]

$$a\ a\ a\ b\ a\ b\ c\ a\ b\ a\ b\ b\ a\ b\ c\ b\ a$$

$$a\ a\ a\ b\ a\ b\ c\ a\ b\ a\ b\ b\ a\ b\ c\ b\ a$$

$$a\ a\ a\ b\ a\ b\ c\ a\ b\ a\ b\ b\ a\ b\ c\ b\ a$$

$$a\ a\ a\ b\ a\ b\ c\ a\ b\ a\ b\ b\ a\ b\ c\ b\ a$$

a_3 b a b c a b a b b a b c b a

a_3 b a b c a b a b b a b c b a

$$a_3 \quad b \; a \; b \; c \; a \; b \; a \quad b_2 \; a \; b \; c \; b \; a$$

$$a_3 \quad b \; a \; b \; c \; a \; b \; a \quad b_2 \; a \; b \; c \; b \; a$$

Equality and Compression of Strings

$$a_3 \quad b \quad d \quad c \quad d \quad a \quad b_2 \quad d \quad c \; b \; a$$

$$a_3 \quad b \quad d \quad c \quad d \quad a \quad b_2 \quad d \quad c \; b \; a$$

$$a_3 \quad b \quad d \quad c \quad d \quad a \quad b_2 \quad d \quad c \quad e$$

$$a_3 \quad b \quad d \quad c \quad d \quad a \quad b_2 \quad d \quad c \quad e$$

Equality and Compression of Strings

$$a_3 \quad b \quad d \quad c \quad d \quad a \quad b_2 \quad d \quad c \quad e$$

$$a_3 \quad b \quad d \quad c \quad d \quad a \quad b_2 \quad d \quad c \quad e$$

Intuition: recompression

- Think of new letters as nonterminals of a grammar
- We build a grammar for both strings, bottom-up.
- Everything is compressed in the same way!

$$a_3 \quad b \quad d \quad c \quad d \quad a \quad b_2 \quad d \quad c \quad e$$

$$a_3 \quad b \quad d \quad c \quad d \quad a \quad b_2 \quad d \quad c \quad e$$

Intuition: recompression

- Think of new letters as nonterminals of a grammar
- We build a grammar for both strings, bottom-up.
- Everything is compressed in the same way!

Comparison with Plandowski's approach

Top-down, creates many problems.

Algorithm: idea

For both solution words choose a pair (or letter) and compress it.

Algorithm: idea

For both solution words choose a pair (or letter) and compress it.

while $U \notin \Sigma$ and $V \notin \Sigma$ **do**
L $\leftarrow$ letters from $S(U) = S(V)$
for choose $ab \in \mathsf{L}^2$ or $a \in \mathsf{L}$ **do**
replace all occurrences of ab in $S(U)$ and $S(V)$
(or replace all occurrences of blocks of a)

Algorithm: idea

For both solution words choose a pair (or letter) and compress it.

while $U \notin \Sigma$ and $V \notin \Sigma$ **do**
    L $\leftarrow$ letters from $S(U) = S(V)$
    for choose $ab \in \mathsf{L}^2$ or $a \in \mathsf{L}$ **do**
        replace all occurrences of ab in $S(U)$ and $S(V)$
        (or replace all occurrences of blocks of a)

How to do this for equations?

Idea at work

Working example

$XbaYb = baaababbab$ has a solution $S(X) = baaa$, $S(Y) = bba$

Working example

$XbaYb = baaababbab$ has a solution $S(X) = baaa$, $S(Y) = bba$

We want to replace pair ba by a new letter c. Then

$$XbaYb{=}baaababbab \quad \text{for } S(X) = baaa\ S(Y) = bba$$

Idea at work

Working example

$XbaYb = baaababbab$ has a solution $S(X) = baaa$, $S(Y) = bba$

We want to replace pair ba by a new letter c. Then

$$
\begin{aligned}
XbaYb &= baaababbab && \text{for } S(X) = baaa \; S(Y) = bba \\
X\,c\,Yb &= c\;aa\;c\;b\;c\;b && \text{for } S'(X) = caa \;\; S'(Y) = bc
\end{aligned}
$$

Working example

$XbaYb = baaababbab$ has a solution $S(X) = baaa$, $S(Y) = bba$

We want to replace pair ba by a new letter c. Then

$$XbaYb{=}baaababbab \quad \text{for } S(X) = baaa\ S(Y) = bba$$
$$X\ c\ Yb{=}\ c\ aa\ c\ b\ c\ b \quad \text{for } S'(X) = caa\ \ S'(Y) = bc$$

And what about replacing ab by d?

$$XbaYb = baaababbab \qquad \text{for } S(X) = baaa\ S(Y) = bba$$

Idea at work

Working example

$XbaYb = baaababbab$ has a solution $S(X) = baaa$, $S(Y) = bba$

We want to replace pair ba by a new letter c. Then

$$\begin{aligned} XbaYb &= baaababbab && \text{for } S(X) = baaa\ S(Y) = bba \\ X\ c\ Yb &= c\ aa\ c\ b\ c\ b && \text{for } S'(X) = caa\ \ S'(Y) = bc \end{aligned}$$

And what about replacing ab by d?

$$XbaYb = baaababbab \qquad \text{for } S(X) = baaa\ S(Y) = bba$$

There is a problem with 'crossing pairs'. We will fix!

Pair types

Definition (Pair types)

Occurrence of ab in a solution word (so for a fixed solution) is

- **explicit** it comes from U or V;
- **implicit** comes solely from $S(X)$;
- **crossing** in other case.

ab is crossing if it has a crossing occurrence, non-crossing otherwise.

Pair types

Definition (Pair types)

Occurrence of ab in a solution word (so for a fixed solution) is

- explicit: it comes from U or V;
- implicit: comes solely from $S(X)$;
- crossing: in other case.

ab is crossing if it has a crossing occurrence, non-crossing otherwise.

$$X \quad baa \;\; Y \;\; b = baaabaabbab \qquad S(X) = baaa \; S(Y) = bba$$

Pair types

Definition (Pair types)

Occurrence of ab in a solution word (so for a fixed solution) is

- **explicit** it comes from U or V;
- **implicit** comes solely from $S(X)$;
- **crossing** in other case.

ab is crossing if it has a crossing occurrence, non-crossing otherwise.

$$X \; baa \; Y \; b = baaabaabbab \qquad S(X) = baaa \; S(Y) = bba$$

$baaa \; baa \; bba \; b = baaabaabbab$	explicit
$baaa \; baa \; bba \; b = baaabaabbab$	implicit
$baaa \; baa \; bba \; b = baaabaabbab$	crossing

Compression of non-crossing pairs

PairComp(a, b)

1: let $c \in \Sigma$ be an unused letter
2: replace each explicit ab in U and V by c

Compression of non-crossing pairs

PairComp(a, b)

1: let $c \in \Sigma$ be an unused letter
2: replace each explicit ab in U and V by c

Lemma

The PairComp(a, b) *properly compresses noncrossing pairs.*

Compression of non-crossing pairs

PairComp(a, b)

1: let $c \in \Sigma$ be an unused letter
2: replace each explicit ab in U and V by c

Lemma

The PairComp(a, b) *properly compresses noncrossing pairs.*

complete: if the old equation has a solution then the new one has

sound: if the new equation has a solution then the old one has

Correctness.

Complete

$S'(U')$ is $S(U)$ with every ab replaced; similarly $S'(V')$:

explicit pairs replaced explicitly

implicit pairs replaced implicitly (in the solution)

crossing there are none

Complete

$S'(U')$ is $S(U)$ with every ab replaced; similarly $S'(V')$:

explicit pairs replaced explicitly

implicit pairs replaced implicitly (in the solution)

crossing there are none

$X \ \ baa\ Y\ b{=}baaabaabbab \quad S(X) = baaa\ S(Y) = bba$

$baaabaabbab{=}baaabaabbab$

Complete

$S'(U')$ is $S(U)$ with every ab replaced; similarly $S'(V')$:

explicit pairs replaced explicitly

implicit pairs replaced implicitly (in the solution)

crossing there are none

$$X \ baa \ Y \ b = baaabaabbab \quad S(X) = baaa \ S(Y) = bba$$

$$baaabaabbab = baaabaabbab$$

$$c \, aa \ c \, ab \ c \, b = c \ aa \ c \ ab \ c \ b$$

$$X \ c \, a \ Y \ b = c \ aa \ c \ ab \ c \ b \quad S'(X) = caa \ S'(Y) = bc$$

Correctness.

Complete

$S'(U')$ is $S(U)$ with every ab replaced; similarly $S'(V')$:

explicit pairs replaced explicitly

implicit pairs replaced implicitly (in the solution)

crossing there are none

Sound

If the new equation is satisfiable: roll back the changes.

Correctness.

Complete

$S'(U')$ is $S(U)$ with every ab replaced; similarly $S'(V')$:

explicit pairs replaced explicitly

implicit pairs replaced implicitly (in the solution)

crossing there are none

Sound

If the new equation is satisfiable: roll back the changes.

$$X \; baa \; Y \; b = baaabaabbab$$

$$c\,aa \; c\,ab\,c\,b = c\,aa\,c\,ab\,c\,b$$

$$X \; c\,a\,Y\,b = c\,aa\,c\,ab\,c\,b \qquad S'(X) = caa \; S'(Y) = bc$$

Correctness.

Complete

$S'(U')$ is $S(U)$ with every ab replaced; similarly $S'(V')$:

explicit pairs replaced explicitly

implicit pairs replaced implicitly (in the solution)

crossing there are none

Sound

If the new equation is satisfiable: roll back the changes.

$$X \ \ baa\ Y\ b = baaabaabbab \quad S(X) = baaa\ S(Y) = bba$$

$$c\,aa\ \ c\ ab\ c\ b = c\ aa\ c\ ab\ c\ b$$

$$X \ \ c\ a\ Y\ b = c\ aa\ c\ ab\ c\ b \quad S'(X) = caa\ S'(Y) = bc$$

Correctness.

Complete

$S'(U')$ is $S(U)$ with every ab replaced; similarly $S'(V')$:

explicit pairs replaced explicitly

implicit pairs replaced implicitly (in the solution)

crossing there are none

Sound

If the new equation is satisfiable: roll back the changes.

$$X \; baa \; Y \; b = baaabaabbab \qquad S(X) = baaa \; S(Y) = bba$$

$$baaabaabbab = baaabaabbab$$

$$c \, aa \; c \, ab \; c \, b = c \; aa \; c \; ab \; c \; b$$

$$X \quad c \; a \; Y \; b = c \; aa \; c \; ab \; c \; b \qquad S'(X) = caa \; S'(Y) = bc$$

Dealing with crossing pairs

ab is a crossing pair

There is X such that $S(X) = bw$ and aX occurs in $U = V$ (or symmetric).

Dealing with crossing pairs

ab is a crossing pair

There is X such that $S(X) = bw$ and aX occurs in $U = V$ (or symmetric).

Uncrossing(a, b)

```
for X ∈ 𝒳 do
    if first letter of S(X) is b then
        replace each occurrence of X by bX                      ▷ Pop
                                                ▷ Change S accordingly
    if S(X) = ϵ then remove X from the equation
                      ▷ perform symmetrically for the last letter and a
```

Dealing with crossing pairs

ab is a crossing pair

There is X such that $S(X) = bw$ and aX occurs in $U = V$ (or symmetric).

Uncrossing(a, b)

1: **for** $X \in \mathcal{X}$ **do**
2: **if** first letter of $S(X)$ is b **then**
3: replace each occurrence of X by bX ▷ Pop ▷ Change S accordingly
4: **if** $S(X) = \epsilon$ **then** remove X from the equation
5: ▷ perform symmetrically for the last letter and a

Lemma

After uncrossing ab is no longer crossing.

ab is a crossing pair

There is X such that $S(X) = bw$ and aX occurs in $U = V$ (or symmetric).

Uncrossing(a, b)

1: **for** $X \in \mathcal{X}$ **do**
2: **if** first letter of $S(X)$ is b **then**
3: replace each occurrence of X by bX ▷ Pop
▷ Change S accordingly
4: **if** $S(X) = \epsilon$ **then** remove X from the equation
5: ▷ perform symmetrically for the last letter and a

Lemma

After uncrossing ab is no longer crossing.

We can compress it.

Uncrossing: example

Uncrossing ab

$$X\ \ baa\ \ Y\ \ b{=}baaabaabbab \quad S(X) = baaa\ S(Y) = bba$$

Uncrossing ab

$$X\ \ baa\ \ Y\ \ b = baaabaabbab \quad S(X) = baaa\ S(Y) = bba$$
$$baaabaa\ bba\ b = baaabaabbab$$

Uncrossing: example

Uncrossing ab

$$X \ \ baa \ \ Y \ \ b = baaabaabbab \qquad S(X) = baaa \ S(Y) = bba$$

$$baaabaa\,bba\,b = baaabaabbab$$

$$bXabaabYab = baaabaabbab \qquad S'(X) = aa \ S'(Y) = b$$

Uncrossing: example

Uncrossing ab

$$X\ \ baa\ \ Y\ \ b = baaabaabbab \quad S(X) = baaa\ S(Y) = bba$$

$$baaabaa\ bba\ b = baaabaabbab$$

$$baaabaab\ ba\ b = baaabaabbab$$

$$bXabaabYab = baaabaabbab \qquad S'(X) = aa\ S'(Y) = b$$

Maximal blocks

Definition (maximal block of a)

When a^ℓ occurs in $S(U) = S(V)$ and cannot be extended.

Maximal blocks

Definition (maximal block of a)

When a^ℓ occurs in $S(U) = S(V)$ and cannot be extended.

Equivalents of pairs.

Maximal blocks

Definition (maximal block of a)

When a^ℓ occurs in $S(U) = S(V)$ and cannot be extended.

Equivalents of pairs.

- Block occurrence can be explicit, implicit or crossing.
- Letter a is crossing (has a crossing block) if there is a crossing block of a.

Maximal blocks

Definition (maximal block of a)

When a^ℓ occurs in $S(U) = S(V)$ and cannot be extended.

Equivalents of pairs.

- Block occurrence can be explicit, implicit or crossing.
- Letter a is crossing (has a crossing block) if there is a crossing block of a.

$$X \quad baa\, Y\, b = baabbaabbb \qquad S(X) = baab\ S(Y) = bb$$

$$b\,aab\ baa\ bb\ b = b\,aa\,bb\,aa\,bbb$$

Maximal blocks

Definition (maximal block of a)

When a^ℓ occurs in $S(U) = S(V)$ and cannot be extended.

Equivalents of pairs.

- Block occurrence can be explicit, implicit or crossing.
- Letter a is crossing (has a crossing block) if there is a crossing block of a.

$$X \quad baa\, Y\, b = baabbaabbb \qquad S(X) = baab\; S(Y) = bb$$
$$baab\, baa\, bb\, b = baabbaabbb$$

Lemma (Length-minimal solutions)

If a^ℓ is a maximal block in a length-minimal solution of $U = V$ then $\ell \leq 2^{c|UV|}$.

Blocks compression

When a has no crossing block

for all maximal blocks a^ℓ of a and $\ell > 1$ **do**
    let $a_\ell \in \Sigma$ be an unused letter
    replace each explicit maximal a^ℓ in $U = V$ by a_ℓ

Blocks compression

When a has no crossing block

1: **for** all maximal blocks a^ℓ of a and $\ell > 1$ **do**
2: let $a_\ell \in \Sigma$ be an unused letter
3: replace each explicit maximal a^ℓ in $U = V$ by a_ℓ

Lemma

The BlockComp(a) *properly compresses noncrossing blocks of* a*.*

Blocks compression

When a has no crossing block

1: **for** all maximal blocks a^ℓ of a and $\ell > 1$ **do**
2: let $a_\ell \in \Sigma$ be an unused letter
3: replace each explicit maximal a^ℓ in $U = V$ by a_ℓ

Lemma

The BlockComp(a) *properly compresses noncrossing blocks of* a.

$$X \;\; baaY baaa = baabbaabbbaaa \qquad S(X) = baab \; S(Y) = bb$$

When a has no crossing block

1: **for** all maximal blocks a^ℓ of a and $\ell > 1$ **do**
2: let $a_\ell \in \Sigma$ be an unused letter
3: replace each explicit maximal a^ℓ in $U = V$ by a_ℓ

Lemma

The BlockComp(a) *properly compresses noncrossing blocks of* a.

$$X \; baaYbaaa = baabbaabbbaaa \qquad S(X) = baab \; S(Y) = bb$$

$$baabbaabbbaaa = baabbaabbbaaa$$

Blocks compression

When a has no crossing block

1: **for** all maximal blocks a^ℓ of a and $\ell > 1$ **do**
2: let $a_\ell \in \Sigma$ be an unused letter
3: replace each explicit maximal a^ℓ in $U = V$ by a_ℓ

Lemma

The BlockComp(a) *properly compresses noncrossing blocks of* a.

$$X \ \ baaYbaaa = baabbaabbbaaa \qquad S(X) = baab \ S(Y) = bb$$

$$baabbaabbbaaa = baabbaabbbaaa$$

$$X \ \ ba_2Yb \ a_3 = ba_2bba_2bbb \ a_3 \qquad S'(X) = ba_2b \ S'(Y) = bb$$

Blocks compression

When a has no crossing block

1: **for** all maximal blocks a^ℓ of a and $\ell > 1$ **do**
2: let $a_\ell \in \Sigma$ be an unused letter
3: replace each explicit maximal a^ℓ in $U = V$ by a_ℓ

Lemma

The BlockComp(a) *properly compresses noncrossing blocks of* a.

$$X \;\; baaYbaaa = baabbaabbbaaa \qquad S(X) = baab \; S(Y) = bb$$
$$baabbaabbbaaa = baabbaabbbaaa$$
$$ba_2bba_2bbb \; a_3 = ba_2bba_2bbb \; a_3$$
$$X \;\; ba_2Yb \; a_3 = ba_2bba_2bbb \; a_3 \qquad S'(X) = ba_2b \; S'(Y) = bb$$

Crossing a-chains?

- Crossing a-chain: similar to crossing ab.

Crossing a-chains?

- Crossing a-chain: similar to crossing ab.
- pop whole a-prefix and a-suffix:
  $S(X) = a^{\ell_X} w a^{r_X}$: change it to $S(X) = w$

Crossing a-chains?

- Crossing a-chain: similar to crossing ab.
- pop whole a-prefix and a-suffix:
  $S(X) = a^{\ell_X} w a^{r_X}$: change it to $S(X) = w$

for $X \in \mathcal{X}$ **do**
  replace each occurrence of X by $a^{\ell_X} X a^{r_X}$ $\triangleright$ $\ell_X, r_X \geq 0$
  $\triangleright$ a^{ℓ_X} and a^{r_X} are the a-prefix and suffix of $S(X)$
  if $S(X) = \epsilon$ **then**
    remove X from the equation

Crossing a-chains?

- Crossing a-chain: similar to crossing ab.
- pop whole a-prefix and a-suffix:
  $S(X) = a^{\ell_X} w a^{r_X}$: change it to $S(X) = w$

for $X \in \mathcal{X}$ **do**
  replace each occurrence of X by $a^{\ell_X} X a^{r_X}$ $\triangleright$ $\ell_X, r_X \geq 0$
  $\triangleright$ a^{ℓ_X} and a^{r_X} are the a-prefix and suffix of $S(X)$
  if $S(X) = \epsilon$ **then**
    remove X from the equation

Lemma

After uncrossing a is no longer crossing.

Algorithm

while $U \notin \Sigma$ and $V \notin \Sigma$ **do**
    L $\leftarrow$ letters from $U = V$
    choose a pair of letters or a block from L
    if it is crossing **then**
        Uncross it
    Compress it

Soundness

If the new equation has a solution, then also the original one had.

Soundness

If the new equation has a solution, then also the original one had.

Just roll back the changes.

Soundness

If the new equation has a solution, then also the original one had.

Just roll back the changes.

$$X \quad baa\ Y\ b = baaabaabbab$$

$$X \quad c\ a\ Y\ b = c\ aa\ c\ ab\ c\ b \qquad S(X) = caa\ S(Y) = bc$$

Soundness

If the new equation has a solution, then also the original one had.

Just roll back the changes.

$$X \ baa \ Y \ b = baaabaabbab$$

$$c\,aa \ c\,ab\,c\,b = c\,aa\,c\,ab\,c\,b$$

$$X \ c\,a\,Y\,b = c\,aa\,c\,ab\,c\,b \qquad S(X) = caa \ S(Y) = bc$$

Soundness

If the new equation has a solution, then also the original one had.

Just roll back the changes.

$$X\ \ baa\ Y\ b{=}baaabaabbab \quad S(X) = baaa\ S(Y) = bba$$

$$c\,aa\ \ c\,ab\ c\,b{=}\ c\ aa\ c\ ab\ c\ b$$
$$X\ \ \ c\ a\ Y\ b{=}\ c\ aa\ c\ ab\ c\ b \qquad S(X) = caa\ S(Y) = bc$$

Soundness

If the new equation has a solution, then also the original one had.

Just roll back the changes.

$$X\ \ baa\ Y\ b = baaabaabbab \qquad S(X) = baaa\ S(Y) = bba$$
$$baaabaabbab = baaabaabbab$$
$$c\,aa\ \ c\,ab\ c\ b = c\ aa\ c\ ab\ c\ b$$
$$X\ \ c\ a\ Y\ b = c\ aa\ c\ ab\ c\ b \qquad S(X) = caa\ S(Y) = bc$$

Completeness

If the equation has the solution, then for some nondeterministic choices the new equation has a corresponding one.

Completeness

If the equation has the solution, then for some nondeterministic choices the new equation has a corresponding one.

Make the choices according to the solution.

Completeness

If the equation has the solution, then for some nondeterministic choices the new equation has a corresponding one.

Make the choices according to the solution.

What about termination?

Termination

We show that

- we stay in $\mathcal{O}(n^2)$ space.
- After each operation the length-minimal solution shortens.

Termination

We show that

- we stay in $\mathcal{O}(n^2)$ space.
- After each operation the length-minimal solution shortens.

So we terminate on positive instances.

Termination

We show that

- we stay in $\mathcal{O}(n^2)$ space.
- After each operation the length-minimal solution shortens.

So we terminate on positive instances.

Lemma

Each compression decreases the length of the length-minimal solution.

Proof.

We perform the compression on the solution word. □

Strategy

Lemma

Compression of a non-crossing pair/block decreases equation's size.

Proof.

Something is compressed in the equation. □

Strategy

Lemma

Compression of a non-crossing pair/block decreases equation's size.

Proof.

Something is compressed in the equation. □

Strategy

- If there is something non-crossing: compress it.
- If there are only crossing: choose one that minimises the equation.

Lemma (Fixed solution)

There are at most $2n$ different crossing pairs and blocks.

Lemma (Fixed solution)

There are at most $2n$ different crossing pairs and blocks.

Each is associated with a side of an occurrence of a variable.

Lemma (Fixed solution)

There are at most $2n$ different crossing pairs and blocks.

Each is associated with a side of an occurrence of a variable.

Lemma (Fixed solution)

Uncrossing introduces at most $2n$ letters to the equation.

Lemma (Fixed solution)

There are at most $2n$ different crossing pairs and blocks.

Each is associated with a side of an occurrence of a variable.

Lemma (Fixed solution)

Uncrossing introduces at most $2n$ letters to the equation.

Each variable pops left and right one letter
for a-chains: it is compressed immediately afterwards.

Lemma (Fixed solution)

There are at most $2n$ different crossing pairs and blocks.

Each is associated with a side of an occurrence of a variable.

Lemma (Fixed solution)

Uncrossing introduces at most $2n$ letters to the equation.

Each variable pops left and right one letter
for a-chains: it is compressed immediately afterwards.

Lemma

There is always some choice to be $\leq 8n^2$.

Lemma (Fixed solution)

There are at most $2n$ different crossing pairs and blocks.

Each is associated with a side of an occurrence of a variable.

Lemma (Fixed solution)

Uncrossing introduces at most $2n$ letters to the equation.

Each variable pops left and right one letter
for a-chains: it is compressed immediately afterwards.

Lemma

There is always some choice to be $\leq 8n^2$.

There are $m \leq 8n^2$ letters and $k \leq 2n$ different crossing blocks/pairs.
Some covers $\geq m/k$ letters.
Its compression removes $\geq m/2k$ letters and introduces $2n$ letters.
We are left with at most

$$(1 - 1/2k) \cdot m + 2n \leq (1 - 1/4n) \cdot 8n^2 + 2n = 8n^2 \quad .$$

Conclusions and Open questions

Conclusions

- The representation can be more important than the combinatorics.

Conclusions and Open questions

Conclusions

- The representation can be more important than the combinatorics.

Open questions

- Are word equations in NP? (Are solutions at most exponential?)
- To which problems can we generalise this approach?

Regular constraints

Regular constraints

For each variable: constraints of the form $X \in R, X \notin R'$

Regular constraints

Regular constraints

For each variable: constraints of the form $X \in R, X \notin R'$

ρ: homomorphism from letters to transition matrices of NFAs
extend also to variables: ρ_X, require $\rho(S(X)) = \rho_X$

Regular constraints

Regular constraints

For each variable: constraints of the form $X \in R, X \notin R'$

ρ: homomorphism from letters to transition matrices of NFAs
extend also to variables: ρ_X, require $\rho(S(X)) = \rho_X$

when w is replaced by c: $\rho(c) \leftarrow \rho(w)$
when X is replaced with wX: $\rho_X \leftarrow \rho'_X$ such that $\rho_X = \rho(w)\rho'_X$
when X i removed: check $\rho_X = \rho(\epsilon)$
(some extra tricks in the analysis)

Space saving

Using parallel compression: length $\mathcal{O}(n) \implies \mathcal{O}(n \log n)$ bits

Using Huffman coding: linear-size (in terms of bits)

Space saving

Using parallel compression: length $\mathcal{O}(n) \implies \mathcal{O}(n \log n)$ bits

Using Huffman coding: linear-size (in terms of bits)
Even if input is Huffman-coded.