Combinatorics on Words

through the Word-Equations-lens

Florin Manea

Georg-August-Universität Göttingen

LMW 2020, 6.7.2020

Combinatorics on Words

Combinatorics on Words...

Wikipedia:
Combinatorics on words is a fairly new field of mathematics, branching from combinatorics, which focuses on the study of words and formal languages. One looks at letters or symbols, and the sequences they form. Combinatorics on words affects various areas of mathematical study, including algebra and computer science. [..]
As time went on, combinatorics on words became useful in the study of algorithms and coding. It led to developments in abstract algebra and answering open questions.

Combinatorics on Words

Combinatorics on Words...

Wikipedia:
Combinatorics on words is a fairly new field of mathematics, branching from combinatorics, which focuses on the study of words and formal languages. One looks at letters or symbols, and the sequences they form. Combinatorics on words affects various areas of mathematical study, including algebra and computer science. [..]
As time went on, combinatorics on words became useful in the study of algorithms and coding. It led to developments in abstract algebra and answering open questions.

... and Stringology

The study of problems, algorithms, and data structures related to (efficient) string processing (Zvi Galil, 1984).

Combinatorics on Words

Combinatorics on Words...

Wikipedia:
Combinatorics on words is a fairly new field of mathematics, branching from combinatorics, which focuses on the study of words and formal languages. One looks at letters or symbols, and the sequences they form. Combinatorics on words affects various areas of mathematical study, including algebra and computer science. [..]
As time went on, combinatorics on words became useful in the study of algorithms and coding. It led to developments in abstract algebra and answering open questions.

... and Stringology

The study of problems, algorithms, and data structures related to (efficient) string processing (Zvi Galil, 1984).

THE References:
M. Lothaire - Combinatorics on Words (1983/1997), Algebraic Combinatorics on Words (2002), Applied Combinatorics on Words (2005).

CombWo via Word Equations

CombWo via Word Equations

- Let $X := \{x, y, z, \ldots\}$ be a (usually infinite) set of variables.

CombWo via Word Equations

- Let $X := \{x, y, z, \ldots\}$ be a (usually infinite) set of variables.
- Let $A := \{\mathtt{a}, \mathtt{b}, \mathtt{c}, \ldots\}$ be a (usually finite) set of terminal symbols.

CombWo via Word Equations

- Let $X := \{x, y, z, \ldots\}$ be a (usually infinite) set of variables.
- Let $A := \{\mathtt{a}, \mathtt{b}, \mathtt{c}, \ldots\}$ be a (usually finite) set of terminal symbols.
- Pattern: $U \in (A \cup X)^*$.

CombWo via Word Equations

- Let $X := \{x, y, z, \ldots\}$ be a (usually infinite) set of variables.
- Let $A := \{\mathtt{a}, \mathtt{b}, \mathtt{c}, \ldots\}$ be a (usually finite) set of terminal symbols.
- Pattern: $U \in (A \cup X)^*$.
- Word: $w \in A^*$; $w = w[1]w[2]\cdots w[n]$;
  Factor: $w[i:j] = w[i]w[i+1]\cdots w[j]$, Prefix: $w[1:i]$, Suffix: $w[i:n]$.

CombWo via Word Equations

- Let $X := \{x, y, z, \ldots\}$ be a (usually infinite) set of variables.
- Let $A := \{\mathtt{a}, \mathtt{b}, \mathtt{c}, \ldots\}$ be a (usually finite) set of terminal symbols.
- Pattern: $U \in (A \cup X)^*$.
- Word: $w \in A^*$; $w = w[1]w[2]\cdots w[n]$;
  Factor: $w[i:j] = w[i]w[i+1]\cdots w[j]$, Prefix: $w[1:i]$, Suffix: $w[i:n]$.

Anatomy of a Word Equation:

- Let $U, V \in (X \cup A)^*$. Then $U = V$ is a word equation.

CombWo via Word Equations

- Let $X := \{x, y, z, \ldots\}$ be a (usually infinite) set of variables.
- Let $A := \{\mathtt{a}, \mathtt{b}, \mathtt{c}, \ldots\}$ be a (usually finite) set of terminal symbols.
- Pattern: $U \in (A \cup X)^*$.
- Word: $w \in A^*$; $w = w[1]w[2]\cdots w[n]$;
  Factor: $w[i:j] = w[i]w[i+1]\cdots w[j]$, Prefix: $w[1:i]$, Suffix: $w[i:n]$.

Anatomy of a Word Equation:

- Let $U, V \in (X \cup A)^*$. Then $U = V$ is a word equation.
- Solutions are substitutions of terminal words for the variables such that the LHS and RHS become identical.

CombWo via Word Equations

- Let $X := \{x, y, z, \ldots\}$ be a (usually infinite) set of variables.
- Let $A := \{\mathtt{a}, \mathtt{b}, \mathtt{c}, \ldots\}$ be a (usually finite) set of terminal symbols.
- Pattern: $U \in (A \cup X)^*$.
- Word: $w \in A^*$; $w = w[1]w[2]\cdots w[n]$;
  Factor: $w[i:j] = w[i]w[i+1]\cdots w[j]$, Prefix: $w[1:i]$, Suffix: $w[i:n]$.

Anatomy of a Word Equation:

- Let $U, V \in (X \cup A)^*$. Then $U = V$ is a word equation.
- Solutions are substitutions of terminal words for the variables such that the LHS and RHS become identical.
- In other words, solutions are terminal-preserving morphisms $h : (X \cup A)^* \to A^*$ such that $h(U) = h(V)$.

CombWo via Word Equations

- Let $X := \{x, y, z, \ldots\}$ be a (usually infinite) set of variables.
- Let $A := \{\mathtt{a}, \mathtt{b}, \mathtt{c}, \ldots\}$ be a (usually finite) set of terminal symbols.
- Pattern: $U \in (A \cup X)^*$.
- Word: $w \in A^*$; $w = w[1]w[2]\cdots w[n]$;
  Factor: $w[i:j] = w[i]w[i+1]\cdots w[j]$, Prefix: $w[1:i]$, Suffix: $w[i:n]$.

Anatomy of a Word Equation:

- Let $U, V \in (X \cup A)^*$. Then $U = V$ is a word equation.
- Solutions are substitutions of terminal words for the variables such that the LHS and RHS become identical.
- In other words, solutions are terminal-preserving morphisms $h : (X \cup A)^* \to A^*$ such that $h(U) = h(V)$.
- Inequation: $U \neq V$. Solutions $h : (X \cup A)^* \to A^*$ such that $h(U) \neq h(V)$.

Word Equations

Anatomy of a Word Equation:

$$\mathtt{a}\ x\ \mathtt{ab}\ y\ \mathtt{ab}\ x\ \mathtt{b} = y\ \mathtt{ba}\ x\ \mathtt{aabb}\ x$$

Word Equations

Anatomy of a Word Equation:

$$\mathtt{a}\ x\ \mathtt{ab}\ y\ \mathtt{ab}\ x\ \mathtt{b} = y\ \mathtt{ba}\ x\ \mathtt{aabb}\ x$$

Word Equations

Anatomy of a Word Equation:

$$\mathtt{a}\ X\ \mathtt{ab}\ Y\ \mathtt{ab}\ X\ \mathtt{b} = Y\ \mathtt{ba}\ X\ \mathtt{aabb}\ X$$

Word Equations

Anatomy of a Word Equation:

$$\mathtt{a}\ X\ \mathtt{ab}\ Y\ \mathtt{ab}\ X\ \mathtt{b} = Y\ \mathtt{ba}\ X\ \mathtt{aabb}\ X$$

Solutions are subtitutions of the variables which equate the two sides.

Word Equations

Anatomy of a Word Equation:

$$\mathtt{a}\ \mathtt{b}\ \mathtt{ab}\ y\ \mathtt{ab}\ \mathtt{b}\ \mathtt{b} \;=\; y\ \mathtt{ba}\ \mathtt{b}\ \mathtt{aabb}\ \mathtt{b}$$

Solutions are subtitutions of the variables which equate the two sides.

Word Equations

Anatomy of a Word Equation:

$$\texttt{a b ab aba ab b b} = \texttt{aba ba b aabb b}$$

Solutions are subtitutions of the variables which equate the two sides.

Word Equations

Anatomy of a Word Equation:

$$\mathtt{a\ b\ ab\ aba\ ab\ b\ b} \;=\; \mathtt{aba\ ba\ b\ aabb\ b}$$

Solution h maps $x \to \mathtt{b}$, $y \to \mathtt{aba}$

Solutions are subtitutions of the variables which equate the two sides.

Word Equations

Examples:

- $\mathtt{a}\ x\ \mathtt{ab}\ x\ \mathtt{b} = \mathtt{ababb}\ x$

Word Equations

Examples:

- $\mathtt{a}\ x\ \mathtt{ab}\ x\ \mathtt{b} = \mathtt{ababb}\ x$
  exactly one solution: $h(x) = \mathtt{b}$

Word Equations

Examples:

- $\mathtt{a}\ x\ \mathtt{ab}\ x\ \mathtt{b} = \mathtt{ababb}\ x$
  exactly one solution: $h(x) = \mathtt{b}$
- $x\ \mathtt{a} = \mathtt{a}\ x$

Word Equations

Examples:

- $\mathtt{a}\ x\ \mathtt{a}\,\mathtt{b}\ x\ \mathtt{b} = \mathtt{a}\,\mathtt{b}\,\mathtt{a}\,\mathtt{b}\,\mathtt{b}\ x$
  exactly one solution: $h(x) = \mathtt{b}$
- $x\ \mathtt{a} = \mathtt{a}\ x$
  solutions of the form: $h(x) = a^i$

Word Equations

Examples:

- $\mathtt{a}\ x\ \mathtt{a}\,\mathtt{b}\ x\ \mathtt{b} = \mathtt{a}\,\mathtt{b}\,\mathtt{a}\,\mathtt{b}\,\mathtt{b}\ x$
  exactly one solution: $h(x) = \mathtt{b}$
- $x\ \mathtt{a} = \mathtt{a}\ x$
  solutions of the form: $h(x) = a^i$
- $x\ y = y\ x$

Word Equations

Examples:

- $\mathtt{a}\ x\ \mathtt{ab}\ x\ \mathtt{b} = \mathtt{ababb}\ x$
  exactly one solution: $h(x) = \mathtt{b}$
- $x\ \mathtt{a} = \mathtt{a}\ x$
  solutions of the form: $h(x) = a^i$
- $x\ y = y\ x$
  solutions of the form: $h(x) = w^i,\ h(y) = w^j$

Word Equations

Examples:

- $\mathtt{a}\ x\ \mathtt{ab}\ x\ \mathtt{b} = \mathtt{ababb}\ x$
  exactly one solution: $h(x) = \mathtt{b}$
- $x\ \mathtt{a} = \mathtt{a}\ x$
  solutions of the form: $h(x) = a^i$
- $x\ y = y\ x$
  solutions of the form: $h(x) = w^i,\ h(y) = w^j$
- $\mathtt{a}\ x\ x = x\ \mathtt{b}\ y$

Word Equations

Examples:

- $\mathtt{a}\ x\ \mathtt{ab}\ x\ \mathtt{b} = \mathtt{ababb}\ x$
  exactly one solution: $h(x) = \mathtt{b}$
- $x\ \mathtt{a} = \mathtt{a}\ x$
  solutions of the form: $h(x) = a^i$
- $x\ y = y\ x$
  solutions of the form: $h(x) = w^i,\ h(y) = w^j$
- $\mathtt{a}\ x\ x = x\ \mathtt{b}\ y$
  no solutions!

The Satisfiability Problem

Decision Problem (Satisfiability of Word Equations):

Given a word equation $U = V$, does there exist a solution h satisfying $h(U) = h(V)$?

Size of input: $|U| + |V|$.
[Similar for systems (sets) of equations and inequations.]

The Satisfiability Problem

Decision Problem (Satisfiability of Word Equations):

Given a word equation $U = V$, does there exist a solution h satisfying $h(U) = h(V)$?

Size of input: $|U| + |V|$.
[Similar for systems (sets) of equations and inequations.]

More General and Vague Problem:

Given a word equation $U = V$, describe in a succinct way (if possible) all solutions h satisfying $h(U) = h(V)$.

The Satisfiability Problem

Decision Problem (Satisfiability of Word Equations):

Given a word equation $U = V$, does there exist a solution h satisfying $h(U) = h(V)$?

Size of input: $|U| + |V|$.
[Similar for systems (sets) of equations and inequations.]

More General and Vague Problem:

Given a word equation $U = V$, describe in a succinct way (if possible) all solutions h satisfying $h(U) = h(V)$.

Observation [Karhumäki, Mignosi, Plandowski, 2000]

For every system of word equations and inequations one can construct an equivalent equation.

The Satisfiability Problem

Volker Diekert: More Than 1700 Years of Word Equations. CAI 2015: 22-28

The Satisfiability Problem

Volker Diekert: More Than 1700 Years of Word Equations. CAI 2015: 22-28

- Satisfiability of word equations was first (explicitly) considered by Markov in an attempt to show that Hilbert's 10th Problem is undecidable.

Martin Davis:

"[..] That problem was once thought to be undecidable. In fact, I actually spent some time long ago trying to prove that!"

The Satisfiability Problem

Volker Diekert: More Than 1700 Years of Word Equations. CAI 2015: 22-28

- Satisfiability of word equations was first (explicitly) considered by Markov in an attempt to show that Hilbert's 10th Problem is undecidable.

Martin Davis:

"[..] That problem was once thought to be undecidable. In fact, I actually spent some time long ago trying to prove that!"

- However, it was eventually shown by Makanin that the satisfiability of word equations is decidable.

Martin Davis:

"[..] I'm sure that you and your colleagues are aware of Makanin's general algorithm for such equations."

The Satisfiability Problem

Volker Diekert: More Than 1700 Years of Word Equations. CAI 2015: 22-28

- Satisfiability of word equations was first (explicitly) considered by Markov in an attempt to show that Hilbert's 10th Problem is undecidable.

Martin Davis:

"[..] That problem was once thought to be undecidable. In fact, I actually spent some time long ago trying to prove that!"

- However, it was eventually shown by Makanin that the satisfiability of word equations is decidable.

Martin Davis:

"[..] I'm sure that you and your colleagues are aware of Makanin's general algorithm for such equations."

- Plandowski later showed that the satisfiabilty problem can be solved in PSPACE, and recently this was improved to linear space via recompression by Jez.

The Satisfiability Problem

Volker Diekert: More Than 1700 Years of Word Equations. CAI 2015: 22-28

- Satisfiability of word equations was first (explicitly) considered by Markov in an attempt to show that Hilbert's 10th Problem is undecidable.

Martin Davis:

"[..] That problem was once thought to be undecidable. In fact, I actually spent some time long ago trying to prove that!"

- However, it was eventually shown by Makanin that the satisfiability of word equations is decidable.

Martin Davis:

"[..] I'm sure that you and your colleagues are aware of Makanin's general algorithm for such equations."

- Plandowski later showed that the satisfiabilty problem can be solved in PSPACE, and recently this was improved to linear space via recompression by Jez.
- On the other hand, it is fairly easy to show that the problem is NP-hard. Whether it is NP-complete remains a major open problem.

The Satisfiability Problem

Whether the satisfiabilty problem is NP-complete remains a major open problem.

One way to attack this problem (also for "fragments" of the theory of word equations).

Theorem (Plandowski, Rytter, ICALP 1998)

Suppose that for a given class of word equations, there exists a polynomial P such that any equation in the class which has a solution, has one whose length is at most $2^{P(n)}$ where n is the length of the equation. Then the satisfiability problem for that class is in NP.

Simple Word Equations?

Simple Word Equations?

String Matching: Let p (pattern) and t (text) be words over the terminal alphabet.

- Consider the equation $x\ p\ y = t$. Does there exist a solution h satisfying $h(x)ph(y) = t$?

Simple Word Equations?

String Matching: Let p (pattern) and t (text) be words over the terminal alphabet.

- Consider the equation $x\ p\ y = t$. Does there exist a solution h satisfying $h(x)ph(y) = t$?
- In other words: does p occur in t?

Simple Word Equations?

String Matching: Let p (pattern) and t (text) be words over the terminal alphabet.

- Consider the equation $x\ p\ y = t$. Does there exist a solution h satisfying $h(x)ph(y) = t$?
- In other words: does p occur in t?

Knuth, Morris, Pratt 1970s; Matiyasevich 1969

We can find all solutions to this equation in $O(|p| + |t|)$ time!

Simple Word Equations?

String Matching: Let p (pattern) and t (text) be words over the terminal alphabet.

- Consider the equation $x\ p\ y = t$. Does there exist a solution h satisfying $h(x)ph(y) = t$?
- In other words: does p occur in t?

Knuth, Morris, Pratt 1970s; Matiyasevich 1969

We can find all solutions to this equation in $O(|p| + |t|)$ time!

Main ideas: pure combinatorial

- Discover (and store) how the prefixes of p can be aligned with themselves: $\pi[i]$ = longest proper border of $p[1 : i]$
  (i.e., prefix of $p[1 : i]$ which is also a suffix of $p[1 : i]$, shorter than i)
- While reading the text t maintain (using the function π) the longest prefix of p which is a suffix of the prefix of t we've seen.

A Special (Simpler?) Type of Equations

Pattern Matching with Variables: Match

Given a pattern U and a word w, solve (find all solutions of) the equation $U = w$.

A Special (Simpler?) Type of Equations

Pattern Matching with Variables: Match

Given a pattern U and a word w, solve (find all solutions of) the equation $U = w$.

... that is

pattern U matches word $w \iff \exists$ substitution $h : h(U) = w$.

A Special (Simpler?) Type of Equations

Pattern Matching with Variables: Match

Given a pattern U and a word w, solve (find all solutions of) the equation $U = w$.

... that is

pattern U matches word $w \iff \exists$ substitution $h : h(U) = w$.

$$U = x_1\, x_2\, x_1\, x_3\, x_2$$

$$w = \mathtt{a\,b\,b\,b\,a\,a\,b\,b\,a\,a\,a\,b\,a\,b\,a}$$

A Special (Simpler?) Type of Equations

Pattern Matching with Variables: Match

Given a pattern U and a word w, solve (find all solutions of) the equation $U = w$.

... that is

pattern U matches word $w \iff \exists$ substitution $h : h(U) = w$.

$$U = \mathtt{a}\,\mathtt{b}\,\mathtt{b}\,x_2\,\mathtt{a}\,\mathtt{b}\,\mathtt{b}\,x_3\,x_2$$

$$w = \mathtt{a}\,\mathtt{b}\,\mathtt{b}\,\mathtt{b}\,\mathtt{a}\,\mathtt{a}\,\mathtt{b}\,\mathtt{b}\,\mathtt{a}\,\mathtt{a}\,\mathtt{a}\,\mathtt{b}\,\mathtt{a}\,\mathtt{b}\,\mathtt{a}$$

A Special (Simpler?) Type of Equations

Pattern Matching with Variables: Match

Given a pattern U and a word w, solve (find all solutions of) the equation $U = w$.

... that is

pattern U matches word $w \iff \exists$ substitution $h : h(U) = w$.

$$U = \mathtt{a}\,\mathtt{b}\,\mathtt{b}\,\mathtt{b}\,\mathtt{a}\,\mathtt{a}\,\mathtt{b}\,\mathtt{b}\,x_3\,\mathtt{b}\,\mathtt{a}$$

$$w = \mathtt{a}\,\mathtt{b}\,\mathtt{b}\,\mathtt{b}\,\mathtt{a}\,\mathtt{a}\,\mathtt{b}\,\mathtt{b}\,\mathtt{a}\,\mathtt{a}\,\mathtt{a}\,\mathtt{b}\,\mathtt{a}\,\mathtt{b}\,\mathtt{a}$$

A Special (Simpler?) Type of Equations

Pattern Matching with Variables: Match

Given a pattern U and a word w, solve (find all solutions of) the equation $U = w$.

... that is

pattern U matches word $w \iff \exists$ substitution $h : h(U) = w$.

$$U = \mathtt{a\,b\,b\,b\,a\,a\,b\,b\,a\,a\,a\,b\,a\,b\,a}$$

$$w = \mathtt{a\,b\,b\,b\,a\,a\,b\,b\,a\,a\,a\,b\,a\,b\,a}$$

A Special (Simpler?) Type of Equations

Pattern Matching with Variables: Match

Given a pattern U and a word w, solve (find all solutions of) the equation $U = w$.

... that is

pattern U matches word $w \iff \exists$ substitution $h : h(U) = w$.

Pattern Matching with Variables: Search

Given a pattern U and a word w, find all solutions of the equation $xUy = w$, where x, y are variables not occurring in U.

Motivation

- learning theory (inductive inference, PAC learning),
- language theory (pattern languages),
- pattern matching (parameterised matching, (generalised) function matching),
- matchtest for regular expressions with backreferences (text editors (grep, emacs), programming language (Perl, Java, Python)),
- string solvers (formal verification),
- database theory,
- bioinformatics.

Results

Matching Problem (Match)

Given a pattern U, a word w. Is $U = w$ satisfiable (i. e., $\exists h : h(U) = w$)?

Results

Matching Problem (Match)

Given a pattern U, a word w. Is $U = w$ satisfiable (i. e., $\exists h : h(U) = w$)?

- Match is (in general) NP-complete, even if non-trivial numerical parameters are restricted (e.g., alphabet size 2, each variable has at most 2 occurrences, etc.).

Results

Matching Problem (Match)

Given a pattern U, a word w. Is $U = w$ satisfiable (i. e., $\exists h : h(U) = w$)?

- Match is (in general) NP-complete, even if non-trivial numerical parameters are restricted (e.g., alphabet size 2, each variable has at most 2 occurrences, etc.).

This is a long way from KMP's solution for $x\ p\ y = t$...

Results

Matching Problem (Match)

Given a pattern U, a word w. Is $U = w$ satisfiable (i. e., $\exists h : h(U) = w$)?

- Match is (in general) NP-complete, even if non-trivial numerical parameters are restricted (e.g., alphabet size 2, each variable has at most 2 occurrences, etc.).

This is a long way from KMP's solution for $x\ p\ y = t$... So what is going on in between?

One-Variable Patterns

Simple extension: Allow (multiple occurrences of) one variable in $p \to$ one-variable pattern U.

One-Variable Patterns

Simple extension: Allow (multiple occurrences of) one variable in $p \rightarrow$ one-variable pattern U.
Example: $U = zzz$ or $U = \texttt{aab}z\texttt{ba}zz$.

One-Variable Patterns

Simple extension: Allow (multiple occurrences of) one variable in $p \to$ one-variable pattern U.
Example: $U = zzz$ or $U = \mathtt{aab}z\mathtt{ba}zz$.

Let U be a one-variable pattern, with the variable z, w a word, and let $r = |U|_z$ and $n = |w|$.

Match ($U = w$) can be solved in linear time.

One-Variable Patterns

Simple extension: Allow (multiple occurrences of) one variable in $p \to$ one-variable pattern U.
Example: $U = zzz$ or $U = \mathtt{aab}z\mathtt{ba}zz$.

Let U be a one-variable pattern, with the variable z, w a word, and let $r = |U|_z$ and $n = |w|$.

Match ($U = w$) can be solved in linear time.

Assume we want to solve Search ($xUy = w$).

One-Variable Patterns

Simple extension: Allow (multiple occurrences of) one variable in $p \to$ one-variable pattern U.
Example: $U = zzz$ or $U = \mathtt{aab}z\mathtt{ba}zz$.

Let U be a one-variable pattern, with the variable z, w a word, and let $r = |U|_z$ and $n = |w|$.

Match ($U = w$) can be solved in linear time.

Assume we want to solve Search ($xUy = w$).

Theorem (Kosolobov, M., Nowotka, SPIRE 2017)

Search *can be solved in* $O(rn)$ *time.*

One-Variable Patterns

Simple extension: Allow (multiple occurrences of) one variable in $p \to$ one-variable pattern U.
Example: $U = zzz$ or $U = \mathtt{aab}z\mathtt{ba}zz$.

Let U be a one-variable pattern, with the variable z, w a word, and let $r = |U|_z$ and $n = |w|$.

Match ($U = w$) can be solved in linear time.

Assume we want to solve Search ($xUy = w$).

Theorem (Kosolobov, M., Nowotka, SPIRE 2017)

Search *can be solved in* $O(rn)$ *time.*

There may be $\Theta(n^2)$ matches of U...

One-Variable Patterns

Simple extension: Allow (multiple occurrences of) one variable in $p \to$ one-variable pattern U.
Example: $U = zzz$ or $U = \mathtt{aab}z\mathtt{ba}zz$.

Let U be a one-variable pattern, with the variable z, w a word, and let $r = |U|_z$ and $n = |w|$.

Match ($U = w$) can be solved in linear time.

Assume we want to solve Search ($xUy = w$).

Theorem (Kosolobov, M., Nowotka, SPIRE 2017)

Search *can be solved in* $O(rn)$ *time.*

There may be $\Theta(n^2)$ matches of U...
We can find a compact representation of all these matches.

More Variables: Regular Patterns

- **Regular Patterns**:
  $|U|_x = 1$, for all variables x occurring in U.
  E. g., $U = \mathrm{ab}x_1x_2\mathrm{b}x_3\mathrm{aaa}x_4\mathrm{b}$.

- More variables, no repetition.

More Variables: Regular Patterns

- **Regular Patterns**:
  $|U|_x = 1$, for all variables x occurring in U.
  E. g., $U = \mathrm{ab}x_1x_2\mathrm{b}x_3\mathrm{aaa}x_4\mathrm{b}$.

- More variables, no repetition.

Theorem (folklore)

$U = w$ for a regular pattern U and a word w is solvable in $\mathcal{O}(|U| + |w|)$.

More Variables: Regular Patterns

- **Regular Patterns**:
  $|U|_x = 1$, for all variables x occurring in U.
  E. g., $U = \mathtt{ab}x_1x_2\mathtt{b}x_3\mathtt{aaa}x_4\mathtt{b}$.

- More variables, no repetition.

Theorem (folklore)

$U = w$ for a regular pattern U and a word w is solvable in $\mathcal{O}(|U| + |w|)$.

Greedy strategy: the variables are just "spacers".

More Variables: Regular Patterns

- **Regular Patterns**:
  $|U|_x = 1$, for all variables x occurring in U.
  E. g., $U = \mathtt{ab}x_1x_2\mathtt{b}x_3\mathtt{aaa}x_4\mathtt{b}$.

- More variables, no repetition.

Theorem (folklore)

$U = w$ for a regular pattern U and a word w is solvable in $\mathcal{O}(|U| + |w|)$.

Greedy strategy: the variables are just "spacers". So use KMP to search greedily for the terminal parts!

More Variables: Regular Patterns

- **Regular Patterns**:
  $|U|_x = 1$, for all variables x occurring in U.
  E. g., $U = \mathtt{ab}x_1x_2\mathtt{b}x_3\mathtt{aaa}x_4\mathtt{b}$.

- More variables, no repetition.

Theorem (folklore)

$U = w$ for a regular pattern U and a word w is solvable in $\mathcal{O}(|U| + |w|)$.

Greedy strategy: the variables are just "spacers". So use KMP to search greedily for the terminal parts!
Search is a bit more involved: it is solvable in $\mathcal{O}(|U| + |w| + occ)$, where occ is the number of matches of U in w.

Combine the two extensions...
k-Repeated Variable Patterns

- **k-Repeated-Variable Patterns**:
  $|\{x \in \mathrm{var}(U) \mid |U|_x \geq 2\}| \leq k$.
  E. g., $U = x_1 \mathrm{ab} x_2 \mathrm{a} x_2 \mathrm{a} x_3 \mathrm{ba} x_2 \mathrm{bb} x_4 x_2 x_5$ is a 1-repeated-variable pattern.

Combine the two extensions...
k-Repeated Variable Patterns

- k-**Repeated-Variable Patterns**:
  $|\{x \in \mathrm{var}(U) \mid |U|_x \geq 2\}| \leq k$.
  E. g., $U = x_1 \mathrm{ab} x_2 \mathrm{a} x_2 \mathrm{a} x_3 \mathrm{ba} x_2 \mathrm{bb} x_4 x_2 x_5$ is a 1-repeated-variable pattern.

Lemma (Fernau, M., Mercaş, Schmid, STACS 2015)

Match *for* 1*-repeated-variable patterns is solvable in* $\mathcal{O}(|w|^2)$.

Theorem

Match *for* k*-repeated-variable patterns is solvable in* $\mathcal{O}\left(\frac{|w|^{2k}}{((k-1)!)^2}\right)$.

Match for k-repeated-variable patterns is $W[1]$-hard w.r.t. parameter k.

Combine the two extensions...
Non-Cross Patterns

- **Non-Cross Patterns**: the pattern has a "regular" structure, but instead of single variables, we have one-variable patterns.
  $U = \ldots x \ldots y \ldots x \ldots$ is not possible.
  E. g., $U = x_1\mathtt{aba}x_1\mathtt{a}x_1x_2x_2\mathtt{ba}x_2x_3x_3\mathtt{bb}x_3\mathtt{a}x_3$

Theorem (Fernau, M., Mercaş, Schmid, STACS 2015)

Match *for non-cross patterns is solvable in* $\mathcal{O}(|w|m\log|w|)$, *where* m *is the number of one-variable blocks of the pattern.*

Same complexity for Search.

Combine the two extensions...
Non-Cross Patterns

- **Non-Cross Patterns**: the pattern has a "regular" structure, but instead of single variables, we have one-variable patterns.
  $U = \ldots x \ldots y \ldots x \ldots$ is not possible.
  E. g., $U = x_1 \mathrm{aba} x_1 \mathrm{a} x_1 x_2 x_2 \mathrm{ba} x_2 x_3 x_3 \mathrm{bb} x_3 \mathrm{a} x_3$

Theorem (Fernau, M., Mercaş, Schmid, STACS 2015)

Match *for non-cross patterns is solvable in* $\mathcal{O}(|w| m \log |w|)$, *where* m *is the number of one-variable blocks of the pattern.*

Same complexity for Search.

Open problems:

More interesting (motivated), better parameters leading to poly-time matching? Faster algorithms? Fine grained complexity?

A General Theory

Let U be a pattern. Let $G = (V, E)$ be a graph with $E = E_1 \cup E_2$ such that:

A General Theory

Let U be a pattern. Let $G = (V, E)$ be a graph with $E = E_1 \cup E_2$ such that:

- V is the set $\{1, 2, \ldots |U|\}$ of positions of U,

A General Theory

Let U be a pattern. Let $G = (V, E)$ be a graph with $E = E_1 \cup E_2$ such that:

- V is the set $\{1, 2, \ldots |U|\}$ of positions of U,
- E_1 consists of edges $(i, i + 1)$ between consecutive positions of U,

A General Theory

Let U be a pattern. Let $G = (V, E)$ be a graph with $E = E_1 \cup E_2$ such that:

- V is the set $\{1, 2, \ldots |U|\}$ of positions of U,
- E_1 consists of edges $(i, i + 1)$ between consecutive positions of U,
- There is a path from i to j using (only) edges from E_2 if and only if the i^{th} and j^{th} positions of U are the same variable.

A General Theory

Let U be a pattern. Let $G = (V, E)$ be a graph with $E = E_1 \cup E_2$ such that:

- V is the set $\{1, 2, \ldots |U|\}$ of positions of U,
- E_1 consists of edges $(i, i + 1)$ between consecutive positions of U,
- There is a path from i to j using (only) edges from E_2 if and only if the i^{th} and j^{th} positions of U are the same variable.

Then G is a "valid U-graph".

Patterns with Bounded Treewidth

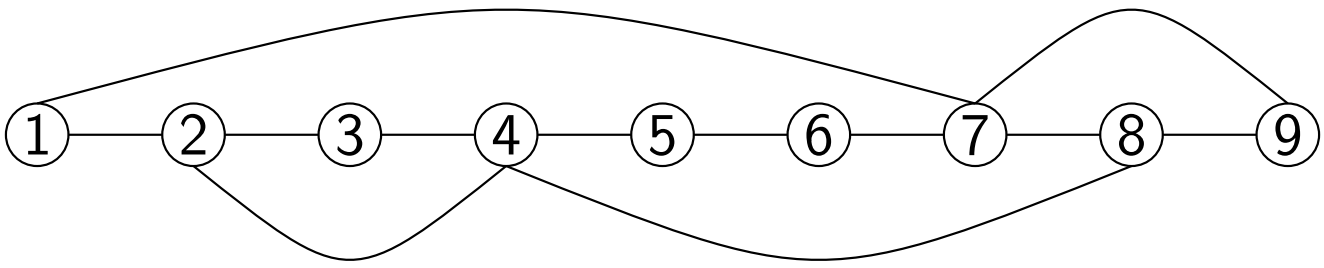
Example: $U = x_1 x_2 \mathtt{a} x_2 \mathtt{a} x_3 x_1 x_2 x_1$

Patterns with Bounded Treewidth

Example: $U = x_1 x_2 \mathtt{a} x_2 \mathtt{a} x_3 x_1 x_2 x_1$

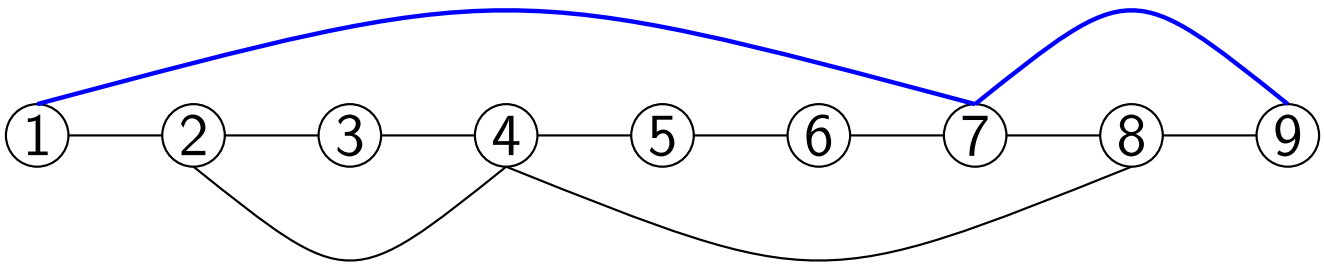
Patterns with Bounded Treewidth

Example: $U = x_1 x_2 \mathtt{a} x_2 \mathtt{a} x_3 x_1 x_2 x_1$



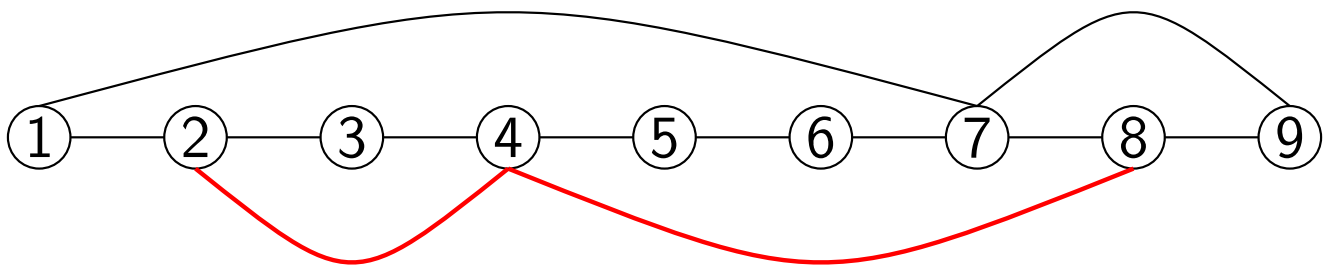
Patterns with Bounded Treewidth

Example: $U = x_1 x_2 a x_2 a x_3 x_1 x_2 x_1$



Patterns with Bounded Treewidth

Example: $U = x_1 x_2 \mathrm{a} x_2 \mathrm{a} x_3 x_1 x_2 x_1$



Patterns with Bounded Treewidth

Reidenbach & Schmid (Inf. Comput. 2014):

A class of patterns C has bounded treewidth if there exists $k \in \mathbb{N}_0$ and a polynomial time computable function mapping each pattern $U \in C$ to a valid U-graph G_U such that G_U has treewidth at most k.

Patterns with Bounded Treewidth

Reidenbach & Schmid (Inf. Comput. 2014):

A class of patterns C has bounded treewidth if there exists $k \in \mathbb{N}_0$ and a polynomial time computable function mapping each pattern $U \in C$ to a valid U-graph G_U such that G_U has treewidth at most k.

- For a class of patterns C with treewidth bounded be a constant k, Match (input U and w) can be solved in $O(|U||w|^{2k+4})$.

Patterns with Bounded Treewidth

Reidenbach & Schmid (Inf. Comput. 2014):

A class of patterns C has bounded treewidth if there exists $k \in \mathbb{N}_0$ and a polynomial time computable function mapping each pattern $U \in C$ to a valid U-graph G_U such that G_U has treewidth at most k.

- For a class of patterns C with treewidth bounded be a constant k, Match (input U and w) can be solved in $O(|U||w|^{2k+4})$.
- All classes presented so far have bounded treewidth...

Patterns with Bounded Treewidth

Reidenbach & Schmid (Inf. Comput. 2014):

A class of patterns C has bounded treewidth if there exists $k \in \mathbb{N}_0$ and a polynomial time computable function mapping each pattern $U \in C$ to a valid U-graph G_U such that G_U has treewidth at most k.

- For a class of patterns C with treewidth bounded be a constant k, Match (input U and w) can be solved in $O(|U||w|^{2k+4})$.
- All classes presented so far have bounded treewidth... but this does not lead to efficient algorithms, rather a method of showing that Match for those classes was in P.

Patterns with Bounded Treewidth

Reidenbach & Schmid (Inf. Comput. 2014):

A class of patterns C has bounded treewidth if there exists $k \in \mathbb{N}_0$ and a polynomial time computable function mapping each pattern $U \in C$ to a valid U-graph G_U such that G_U has treewidth at most k.

- For a class of patterns C with treewidth bounded be a constant k, Match (input U and w) can be solved in $O(|U||w|^{2k+4})$.
- All classes presented so far have bounded treewidth... but this does not lead to efficient algorithms, rather a method of showing that Match for those classes was in P.

Can we match efficiently classes of patterns whose treewidth is unbounded?

Generalized Repetitions

- Let U and V be a patterns. We say that U is a generalized repetition of V if the skeleton of U (all terminals removed from U) is a repetition of the skeleton of V.

  Example: $\mathtt{a}\mathtt{a}x_1\mathtt{a}\mathtt{b}x_2 \cdot \mathtt{a}\mathtt{b}x_1\mathtt{b}\mathtt{a}\mathtt{b}x_2 \cdot \mathtt{a}\mathtt{a}x_1\mathtt{b}\mathtt{b}\mathtt{a}x_2$

Matching Generalized Repetitions of Regular Patterns

Theorem (Day, Fleischmann, M., Nowotka, Schmid, DLT 2018)

We can solve Match *for a generalized repetition of a regular pattern U with m variables and a word w of length n in $O(nm)$ time.*

Theorem

The class of generalized repetitions of regular patterns has unbounded treewidth.

Matching Generalized Repetitions of Regular Patterns

Theorem (Day, Fleischmann, M., Nowotka, Schmid, DLT 2018)

We can solve Match *for a generalized repetition of a regular pattern U with m variables and a word w of length n in $O(nm)$ time.*

Theorem

The class of generalized repetitions of regular patterns has unbounded treewidth.

Can we match efficiently **even more** classes of patterns whose treewidth is unbounded?
See:
Freydenberger, Peterfreund: Finite models and the theory of concatenation. Arxiv 2019.
A new approach to {word equations (the theory of concatenation), pattern matching, document spanners} based on finite model theory (and motivated by data bases!)

Open

Can we explore in a meaningful way the graph-connection? State-of-the-art approximation algorithm for cutwidth obtained this way [Casel et al, ICALP 2019].

Pattern Avoidability

Pattern Avoidability

U pattern, x a new variable.

Is U avoidable?

Given a pattern U does there exist an infinite word ω such that the equation $xU = w$ is unsatisfiable for all prefixes w of ω.

Pattern Avoidability

U pattern, x a new variable.

Is U avoidable?

Given a pattern U does there exist an infinite word ω such that the equation $xU = w$ is unsatisfiable for all prefixes w of ω.

An example:

Thue (1906): Cubes are avoidable

The pattern $U = zzz$ is avoided by the infinite word $0110100110010110\ldots$, generated by applying iteratively the morphism $h(0) = 01$ and $h(1) = 10$ to 0.
That is $U = zzz$ is avoided by $\lim_{i\to\infty} h^i(0)$.

Squares xx, overlaps $yxyxy$, etc. are also avoidable.

Pattern Avoidability

U pattern, x a new variable.

Is U avoidable?

Given a pattern U does there exist an infinite word ω such that the equation $xU = w$ is unsatisfiable for all prefixes w of ω.

A general theory

Zimin words: $Z_1 = x_1$ and $Z_{n+1} = Z_n x_{n+1} Z_n$ for $n \geq 1$.
A pattern U over n distinct pattern variables is unavoidable if and only if the pattern U matches a factor of the n-th Zimin pattern Z_n.

Periodicity Enforcing Equations

Other “simple" word equations?

Periodicity Enforcing Equations

Other “simple" word equations? Like some equations where it is clear which are the solution-sets.

Periodicity Enforcing Equations

Other “simple" word equations? Like some equations where it is clear which are the solution-sets.
The theorems of Lyndon and Schützenberger.

Theorem

- *The solutions of the equation* $x^i = y^j$, *with* $i, j > 0$, *are* $x = u^k$ *and* $y = u^\ell$ *for some* $u \in A^+$ *and* $k, \ell \geq 0$.
- *The solutions of the equation* $xy = yx$ *are* $x = u^k$ *and* $y = u^\ell$ *for some* $u \in A^+$ *and* $k, \ell \geq 0$.

Theorem

The solutions of the equation $x^i y^j = z^k$, *with* $i, j\ell \geq 2$, *are* $x = u^e$, $y = u^f$, $z = u^g$, *for some* $u \in A^*$ *and* $e, f, g \geq 0$.

Theorem

The solutions of the equation $xy = yz$ *are* $x = uv$, $z = vu$, $y = (uv)^e u$, *for* $u, v \in A^*$ *and* $e \geq 0$.

Periodicity Enforcing Equations

Theorem (folklore)

The system $\{xx = yxz, y \neq \epsilon, z \neq \epsilon\}$, *has only solutions* $x = u^e$, $y = u^f$, $z = u^g$, *for some* $u \in A^*$ *and* $e, f, g \geq 0$.

Periodicity Enforcing Equations

Theorem (folklore)

The system $\{xx = yxz, y \neq \epsilon, z \neq \epsilon\}$, *has only solutions* $x = u^e$, $y = u^f$, $z = u^g$, *for some* $u \in A^*$ *and* $e, f, g \geq 0$.

Theorem (Saarela, STACS 2017)

The system $x_0^{k_i} = x_1^{k_i} \dots x_n^{k_i}$ *with* $i \in \{1, 2, 3\}$ *and* k_1, k_2, k_3 *distinct positive integers, has only solutions* $x_j = t^{\ell_j}$, *where* $t \in A^*$ *and* $\ell_j \geq 0$.

Periodicity Enforcing Equations

Theorem (folklore)

The system $\{xx = yxz, y \neq \epsilon, z \neq \epsilon\}$, *has only solutions* $x = u^e$, $y = u^f$, $z = u^g$, *for some* $u \in A^*$ *and* $e, f, g \geq 0$.

Theorem (Saarela, STACS 2017)

The system $x_0^{k_i} = x_1^{k_i} \dots x_n^{k_i}$ *with* $i \in \{1, 2, 3\}$ *and* k_1, k_2, k_3 *distinct positive integers, has only solutions* $x_j = t^{\ell_j}$, *where* $t \in A^*$ *and* $\ell_j \geq 0$.

All constant-free equations have periodic solutions! But do constant-free equations also admit non-periodic solutions?

Periodicity Enforcing Equations

Theorem (folklore)

The system $\{xx = yxz, y \neq \epsilon, z \neq \epsilon\}$, *has only solutions* $x = u^e$, $y = u^f$, $z = u^g$, *for some* $u \in A^*$ *and* $e, f, g \geq 0$.

Theorem (Saarela, STACS 2017)

The system $x_0^{k_i} = x_1^{k_i} \dots x_n^{k_i}$ *with* $i \in \{1, 2, 3\}$ *and* k_1, k_2, k_3 *distinct positive integers, has only solutions* $x_j = t^{\ell_j}$, *where* $t \in A^*$ *and* $\ell_j \geq 0$.

All constant-free equations have periodic solutions! But do constant-free equations also admit non-periodic solutions?

$x^2 = yzy$ has the solution $x = aba$, $y = a$, $z = baab$.

Periodicity Enforcing Equations

Theorem (folklore)

The system $\{xx = yxz, y \neq \epsilon, z \neq \epsilon\}$, *has only solutions* $x = u^e$, $y = u^f$, $z = u^g$, *for some* $u \in A^*$ *and* $e, f, g \geq 0$.

Theorem (Saarela, STACS 2017)

The system $x_0^{k_i} = x_1^{k_i} \dots x_n^{k_i}$ *with* $i \in \{1, 2, 3\}$ *and* k_1, k_2, k_3 *distinct positive integers, has only solutions* $x_j = t^{\ell_j}$, *where* $t \in A^*$ *and* $\ell_j \geq 0$.

All constant-free equations have periodic solutions! But do constant-free equations also admit non-periodic solutions?

$x^2 = yzy$ has the solution $x = aba$, $y = a$, $z = baab$.

Theorem (Saarela, ICALP 2020)

Deciding whether a given constant-free equation has a nonperiodic solution is NP-hard.

On Solving Word Equations with Simple Structure

Structural restrictions made sense for pattern matching.
Which are the structurally simplest word-equations, for each the satisfiability is still hard? (M., Nowotka, Schmid, DLT 2016)

On Solving Word Equations with Simple Structure

Structural restrictions made sense for pattern matching.
Which are the structurally simplest word-equations, for each the satisfiability is still hard? (M., Nowotka, Schmid, DLT 2016)

- Equations with one or two (repeated) variables can be solved in poly-time (see Jez, ICALP 2013).

On Solving Word Equations with Simple Structure

Structural restrictions made sense for pattern matching.
Which are the structurally simplest word-equations, for each the satisfiability is still hard? (M., Nowotka, Schmid, DLT 2016)

- Equations with one or two (repeated) variables can be solved in poly-time (see Jez, ICALP 2013). Open: constant number $(3, 4, ..)$ of variables?

On Solving Word Equations with Simple Structure

Structural restrictions made sense for pattern matching.
Which are the structurally simplest word-equations, for each the satisfiability is still hard? (M., Nowotka, Schmid, DLT 2016)

- Equations with one or two (repeated) variables can be solved in poly-time (see Jez, ICALP 2013). Open: constant number $(3, 4, ..)$ of variables?
- Strictly regular ordered equations:
  $w_1 x_1 w_2 x_2 \cdots w_n x_n w_{n+1} = v_1 x_1 v_2 x_2 \cdots v_n x_n v_{n+1}$.

On Solving Word Equations with Simple Structure

Structural restrictions made sense for pattern matching.
Which are the structurally simplest word-equations, for each the satisfiability is still hard? (M., Nowotka, Schmid, DLT 2016)

- Equations with one or two (repeated) variables can be solved in poly-time (see Jez, ICALP 2013). Open: constant number $(3, 4, ..)$ of variables?
- Strictly regular ordered equations:
  $w_1 x_1 w_2 x_2 \cdots w_n x_n w_{n+1} = v_1 x_1 v_2 x_2 \cdots v_n x_n v_{n+1}$.
- (Day, M. Nowotka, MFCS 2017) Satisfiability of strictly regular ordered equations is NP-complete!

On Solving Word Equations with Simple Structure

Structural restrictions made sense for pattern matching.
Which are the structurally simplest word-equations, for each the satisfiability is still hard? (M., Nowotka, Schmid, DLT 2016)

- Equations with one or two (repeated) variables can be solved in poly-time (see Jez, ICALP 2013). Open: constant number $(3, 4, ..)$ of variables?
- Strictly regular ordered equations:
  $w_1 x_1 w_2 x_2 \cdots w_n x_n w_{n+1} = v_1 x_1 v_2 x_2 \cdots v_n x_n v_{n+1}$.
- (Day, M. Nowotka, MFCS 2017) Satisfiability of strictly regular ordered equations is NP-complete!
- Can we now extend the NP-containment result?

On Solving Word Equations with Simple Structure

Structural restrictions made sense for pattern matching.
Which are the structurally simplest word-equations, for each the satisfiability is still hard? (M., Nowotka, Schmid, DLT 2016)

- Equations with one or two (repeated) variables can be solved in poly-time (see Jez, ICALP 2013). Open: constant number $(3, 4, ..)$ of variables?
- Strictly regular ordered equations:
  $w_1 x_1 w_2 x_2 \cdots w_n x_n w_{n+1} = v_1 x_1 v_2 x_2 \cdots v_n x_n v_{n+1}$.
- (Day, M. Nowotka, MFCS 2017) Satisfiability of strictly regular ordered equations is NP-complete!
- Can we now extend the NP-containment result?
- (Day, M., Nowotka, MFCS 2019): regular reversed equations.

On Solving Word Equations with Simple Structure

Structural restrictions made sense for pattern matching.
Which are the structurally simplest word-equations, for each the satisfiability is still hard? (M., Nowotka, Schmid, DLT 2016)

- Equations with one or two (repeated) variables can be solved in poly-time (see Jez, ICALP 2013). Open: constant number $(3, 4, ..)$ of variables?
- Strictly regular ordered equations:
  $w_1 x_1 w_2 x_2 \cdots w_n x_n w_{n+1} = v_1 x_1 v_2 x_2 \cdots v_n x_n v_{n+1}$.
- (Day, M. Nowotka, MFCS 2017) Satisfiability of strictly regular ordered equations is NP-complete!
- Can we now extend the NP-containment result?
- (Day, M., Nowotka, MFCS 2019): regular reversed equations.
- The satisfiability of regular equations (both sides are regular patterns) is in NP! (Day, M., ICALP 2020)

On Solving Word Equations with Simple Structure

Structural restrictions made sense for pattern matching.
Which are the structurally simplest word-equations, for each the satisfiability is still hard? (M., Nowotka, Schmid, DLT 2016)

- Equations with one or two (repeated) variables can be solved in poly-time (see Jez, ICALP 2013). Open: constant number $(3, 4, ..)$ of variables?
- Strictly regular ordered equations:
  $w_1 x_1 w_2 x_2 \cdots w_n x_n w_{n+1} = v_1 x_1 v_2 x_2 \cdots v_n x_n v_{n+1}$.
- (Day, M. Nowotka, MFCS 2017) Satisfiability of strictly regular ordered equations is NP-complete!
- Can we now extend the NP-containment result?
- (Day, M., Nowotka, MFCS 2019): regular reversed equations.
- The satisfiability of regular equations (both sides are regular patterns) is in NP! (Day, M., ICALP 2020)

Quadratic equations (Diekert, Robson, 1999)

Is the satisfiability of quadratic word equations (a variable occurs at most 2 times) is in NP?

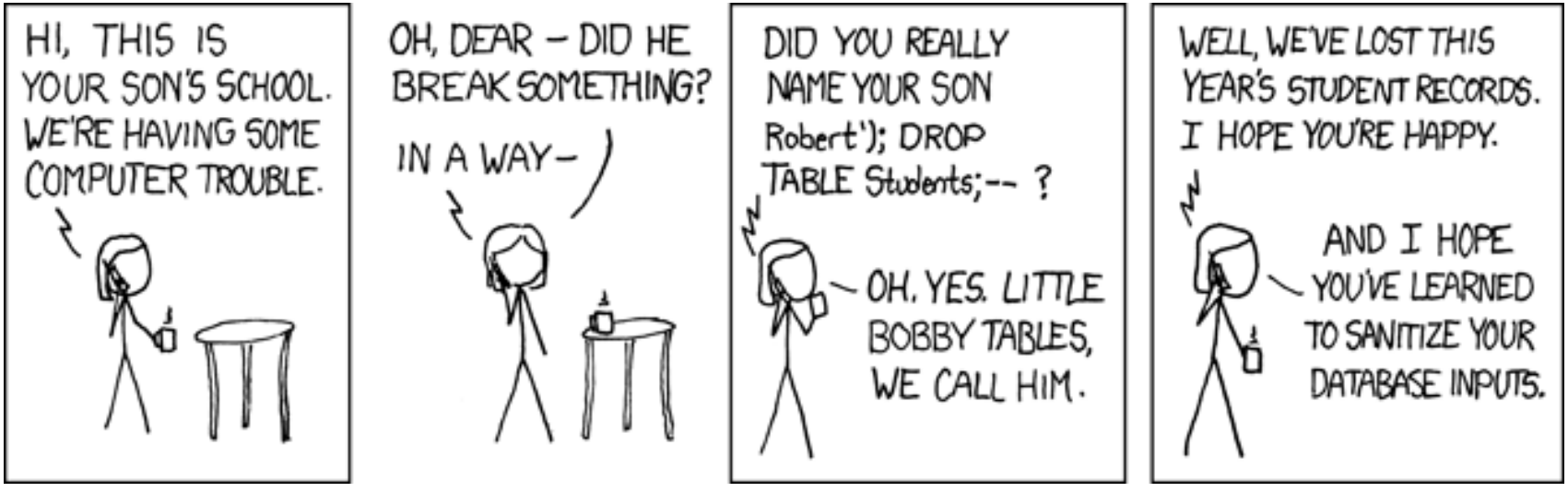
The Satisfiability Problem - Practice

More General Decision Problem(s):

Given a word equation $U = V$, does there exist a solution h satisfying $h(U) = h(V)$ and such that h satisfies some additional properties or constraints?

Motivation

- There are many reasons to care about solving word equations, with applications in various areas from algebra to database theory.
- However, even the linear space recompression algorithm is massively non-deterministic, and thus solving word equations in practice remains largely infeasible.
- Nevertheless, in recent years, there has been elevated interest in solving word equations from the point of view of verification and e-security.



Source: XKCD

Motivation

Source: XKCD

- Several SMT-solvers for strings are being developed which incorporate (incomplete) algorithms for solving word equations.
- Usually, however, it is desirable to be able to look for solutions satisfying some additional conditions, such as comparisons of the lengths of (images of) variables, restricting the (images of) variables to regular languages etc.

Motivation

- So what can we say about these extended theories of word equations? Does the satisfiability problem remain decidable? What about more complex questions (e.g. whether all solutions have a specific form)?

Motivation

- So what can we say about these extended theories of word equations? Does the satisfiability problem remain decidable? What about more complex questions (e.g. whether all solutions have a specific form)?
- In general, these questions need not be decidable.

Motivation

- So what can we say about these extended theories of word equations? Does the satisfiability problem remain decidable? What about more complex questions (e.g. whether all solutions have a specific form)?
- In general, these questions need not be decidable.
- In fact, many simple extensions lead immediately to undecidability. Some lead to decidability.
  E.g., subword-constraints lead to undecidability (Halfon, Schnoebelen, Zetzsche, LICS 2017).
  See (Day et al. RP 2018) for a longer discussion.

Motivation

- So what can we say about these extended theories of word equations? Does the satisfiability problem remain decidable? What about more complex questions (e.g. whether all solutions have a specific form)?
- In general, these questions need not be decidable.
- In fact, many simple extensions lead immediately to undecidability. Some lead to decidability.
  E.g., subword-constraints lead to undecidability (Halfon, Schnoebelen, Zetzsche, LICS 2017).
  See (Day et al. RP 2018) for a longer discussion.
- One particularly interesting case is solving word equations modulo length constraints, whose decidability status has been open for many years.

Regular Constraints

Example:

$$x\ \mathtt{a}\ y\ x\ \mathtt{b} = y\ y\ \mathtt{ab}\ x$$

$$L_x = \mathtt{a}^+\mathtt{b}^+ \quad L_y = \mathtt{b}^+\mathtt{a}^+$$

Regular Constraints

Example:

$$x\ \mathtt{a}\ y\ x\ \mathtt{b} = y\ y\ \mathtt{ab}\ x$$

$$L_x = \mathtt{a}^+\mathtt{b}^+ \quad L_y = \mathtt{b}^+\mathtt{a}^+$$

No solutions!

Regular Constraints

Example:

$$x\ \mathtt{a}\ y\ x\ \mathtt{b} = y\ y\ \mathtt{ab}\ x$$

$$L_x = \mathtt{a}^+\mathtt{b}^+ \quad L_y = \mathtt{b}^+\mathtt{a}^+$$

No solutions!

- Satisfiability of word equations with regular constraints is PSPACE-complete when the constraints are given by NFAs or DFAs.

Length Constraints

- Since solving systems of quadratic Diophantine equations is in general undecidable, we restrict to linear length constraints.

Length Constraints

- Since solving systems of quadratic Diophantine equations is in general undecidable, we restrict to linear length constraints.
- Restricting to only direct equality in length constraints (i.e., $|h(x)| = 2|h(y)| + |h(z)|$ is not allowed, but $|h(x)| = |h(y)|$ yes) is as powerful as allowing arbitrary systems of linear length equations.

Length Constraints

- Since solving systems of quadratic Diophantine equations is in general undecidable, we restrict to linear length constraints.
- Restricting to only direct equality in length constraints (i.e., $|h(x)| = 2|h(y)| + |h(z)|$ is not allowed, but $|h(x)| = |h(y)|$ yes) is as powerful as allowing arbitrary systems of linear length equations.
- Decidabillity of satisfiability for word equations with linear length constraints is an open problem dating back to 1968.
- Decidabillity of satisfiability for regular/quadratic word equations with linear length constraints is also open! (see, e.g., Lin, Majumdar, ATVA 2018)

Even More Word Equations

- Expressibility:
  Karhumäki, Mignosi, Plandowski: The expressibility of languages and relations by word equations. J. ACM (2000)
- Independent Systems:
  Saarela: Independent Systems of Word Equations: From Ehrenfeucht to Eighteen. WORDS 2019
  Nowotka, Saarela: An Optimal Bound on the Solution Sets of One-Variable Word Equations and its Consequences. ICALP 2018

Even More Word Equations

- Expressibility:
  Karhumäki, Mignosi, Plandowski: The expressibility of languages and relations by word equations. J. ACM (2000)
- Independent Systems:
  Saarela: Independent Systems of Word Equations: From Ehrenfeucht to Eighteen. WORDS 2019
  Nowotka, Saarela: An Optimal Bound on the Solution Sets of One-Variable Word Equations and its Consequences. ICALP 2018
- This week @ ICALP:
  Aleksi Saarela: Hardness Results for Constant-Free Pattern Languages and Word Equations
  Joel D. Day, M.: On the Structure of Solution Sets to Regular Word Equations.

The End

Thank you!