Identities of Kauffman monoids: finite axiomatization and algorithms

Mikhail Volkov
(with Karl Auinger, Yuzhu Chen, Xun Hu, Yanfeng Luo, Nikita Kitov)

Ural Federal University, Ekaterinburg, Russia

Outline

- Checking identities
- Kauffman monoids
- Checking identities in $\mathcal{K}_3$
- Checking identities in $\mathcal{K}_4$

Outline

- Checking identities
- Kauffman monoids
- Checking identities in $\mathcal{K}_3$
- Checking identities in $\mathcal{K}_4$

Outline

- Checking identities
- Kauffman monoids
- Checking identities in $\mathcal{K}_3$
- Checking identities in $\mathcal{K}_4$

Outline

- Checking identities
- Kauffman monoids
- Checking identities in $\mathcal{K}_3$
- Checking identities in $\mathcal{K}_4$

Identities

I assume that the idea of an identity or a law is familiar.

Here we deal with semigroup identities; such an identity is just an arbitrary pair of semigroup words, traditionally written as a formal equality. We will write identities using the sign $\bumpeq$, so that the pair (w, w') is written as $w \bumpeq w'$, and reserve the usual equality sign $=$ for 'genuine' equalities.

For a semigroup word w, the set of all letters that occur in w is denoted $\mathrm{alph}(w)$. For instance, $\mathrm{alph}(xy^2zxzy^2x) = \{x, y, z\}$.

Let $\mathcal{S}$ be a semigroup, $w \bumpeq w'$ an identity, and let $X := \mathrm{alph}(ww')$. $\mathcal{S}$ satisfies $w \bumpeq w'$ if $w\varphi = w'\varphi$ for every morphism $\varphi\colon X^+ \to \mathcal{S}$. Observe that φ is uniquely determined by the substitution $\varphi|_X$ so that the definition amounts to saying that every substitution of elements in $\mathcal{S}$ for letters in X yields equal values to w and w'.

Identities

I assume that the idea of an identity or a law is familiar.

Here we deal with semigroup identities; such an identity is just an arbitrary pair of semigroup words, traditionally written as a formal equality. We will write identities using the sign $\bumpeq$, so that the pair (w, w') is written as $w \bumpeq w'$, and reserve the usual equality sign $=$ for 'genuine' equalities.

For a semigroup word w, the set of all letters that occur in w is denoted $\mathrm{alph}(w)$. For instance, $\mathrm{alph}(xy^2zxzy^2x) = \{x, y, z\}$.

Let $\mathcal{S}$ be a semigroup, $w \bumpeq w'$ an identity, and let $X := \mathrm{alph}(ww')$. $\mathcal{S}$ satisfies $w \bumpeq w'$ if $w\varphi = w'\varphi$ for every morphism $\varphi\colon X^+ \to \mathcal{S}$. Observe that φ is uniquely determined by the substitution $\varphi|_X$ so that the definition amounts to saying that every substitution of elements in $\mathcal{S}$ for letters in X yields equal values to w and w'.

Identities

I assume that the idea of an identity or a law is familiar.

Here we deal with semigroup identities; such an identity is just an arbitrary pair of semigroup words, traditionally written as a formal equality. We will write identities using the sign $\bumpeq$, so that the pair (w, w') is written as $w \bumpeq w'$, and reserve the usual equality sign $=$ for 'genuine' equalities.

For a semigroup word w, the set of all letters that occur in w is denoted $\mathrm{alph}(w)$. For instance, $\mathrm{alph}(xy^2zxzy^2x) = \{x, y, z\}$.

Let $\mathcal{S}$ be a semigroup, $w \bumpeq w'$ an identity, and let $X := \mathrm{alph}(ww')$. $\mathcal{S}$ satisfies $w \bumpeq w'$ if $w\varphi = w'\varphi$ for every morphism $\varphi\colon X^+ \to \mathcal{S}$. Observe that φ is uniquely determined by the substitution $\varphi|_X$ so that the definition amounts to saying that every substitution of elements in $\mathcal{S}$ for letters in X yields equal values to w and w'.

Identities

I assume that the idea of an identity or a law is familiar.

Here we deal with semigroup identities; such an identity is just an arbitrary pair of semigroup words, traditionally written as a formal equality. We will write identities using the sign $\bumpeq$, so that the pair (w, w') is written as $w \bumpeq w'$, and reserve the usual equality sign $=$ for 'genuine' equalities.

For a semigroup word w, the set of all letters that occur in w is denoted $\mathrm{alph}(w)$. For instance, $\mathrm{alph}(xy^2zxzy^2x) = \{x, y, z\}$.

Let $\mathcal{S}$ be a semigroup, $w \bumpeq w'$ an identity, and let $X := \mathrm{alph}(ww')$. $\mathcal{S}$ satisfies $w \bumpeq w'$ if $w\varphi = w'\varphi$ for every morphism $\varphi\colon X^+ \to \mathcal{S}$. Observe that φ is uniquely determined by the substitution $\varphi|_X$ so that the definition amounts to saying that every substitution of elements in $\mathcal{S}$ for letters in X yields equal values to w and w'.

Identities

I assume that the idea of an identity or a law is familiar.

Here we deal with semigroup identities; such an identity is just an arbitrary pair of semigroup words, traditionally written as a formal equality. We will write identities using the sign $\bumpeq$, so that the pair (w, w') is written as $w \bumpeq w'$, and reserve the usual equality sign $=$ for 'genuine' equalities.

For a semigroup word w, the set of all letters that occur in w is denoted $\mathrm{alph}(w)$. For instance, $\mathrm{alph}(xy^2zxzy^2x) = \{x, y, z\}$.

Let $\mathcal{S}$ be a semigroup, $w \bumpeq w'$ an identity, and let $X := \mathrm{alph}(ww')$. $\mathcal{S}$ satisfies $w \bumpeq w'$ if $w\varphi = w'\varphi$ for every morphism $\varphi\colon X^+ \to \mathcal{S}$. Observe that φ is uniquely determined by the substitution $\varphi|_X$ so that the definition amounts to saying that every substitution of elements in $\mathcal{S}$ for letters in X yields equal values to w and w'.

Identities

I assume that the idea of an identity or a law is familiar.

Here we deal with semigroup identities; such an identity is just an arbitrary pair of semigroup words, traditionally written as a formal equality. We will write identities using the sign $\bumpeq$, so that the pair (w, w') is written as $w \bumpeq w'$, and reserve the usual equality sign $=$ for 'genuine' equalities.

For a semigroup word w, the set of all letters that occur in w is denoted $\mathrm{alph}(w)$. For instance, $\mathrm{alph}(xy^2zxzy^2x) = \{x, y, z\}$.

Let $\mathcal{S}$ be a semigroup, $w \bumpeq w'$ an identity, and let $X := \mathrm{alph}(ww')$. $\mathcal{S}$ satisfies $w \bumpeq w'$ if $w\varphi = w'\varphi$ for every morphism $\varphi\colon X^+ \to \mathcal{S}$. Observe that φ is uniquely determined by the substitution $\varphi|_X$ so that the definition amounts to saying that every substitution of elements in $\mathcal{S}$ for letters in X yields equal values to w and w'.

Identities

I assume that the idea of an identity or a law is familiar.

Here we deal with semigroup identities; such an identity is just an arbitrary pair of semigroup words, traditionally written as a formal equality. We will write identities using the sign $\bumpeq$, so that the pair (w, w') is written as $w \bumpeq w'$, and reserve the usual equality sign $=$ for 'genuine' equalities.

For a semigroup word w, the set of all letters that occur in w is denoted $\mathrm{alph}(w)$. For instance, $\mathrm{alph}(xy^2zxzy^2x) = \{x, y, z\}$.

Let $\mathcal{S}$ be a semigroup, $w \bumpeq w'$ an identity, and let $X := \mathrm{alph}(ww')$. $\mathcal{S}$ satisfies $w \bumpeq w'$ if $w\varphi = w'\varphi$ for every morphism $\varphi\colon X^+ \to \mathcal{S}$. Observe that φ is uniquely determined by the substitution $\varphi|_X$ so that the definition amounts to saying that every substitution of elements in $\mathcal{S}$ for letters in X yields equal values to w and w'.

Checking Identities

Given a semigroup $\mathcal{S}$, its identity checking problem, denoted $\textsc{Check-Id}(\mathcal{S})$, is a combinatorial decision problem whose instance is a semigroup identity $w \bumpeq w'$; the answer to the instance $w \bumpeq w'$ is "YES" if $\mathcal{S}$ satisfies $w \bumpeq w'$ and "NO" otherwise.

We stress that here $\mathcal{S}$ is fixed and it is the identity $w \bumpeq w'$ that serves as the input so that the time/space complexity should be measured in terms of the size of the identity, i.e., in terms of $|ww'|$.

For a finite semigroup, the identity checking problem is always decidable. Indeed, if $\mathcal{S}$ is finite, then for every identity $w \bumpeq w'$, there are only finitely many substitutions of elements in $\mathcal{S}$ for letters in $X := \mathrm{alph}(ww')$, and one can consecutively calculate the values of w and w' under each of these substitutions.

This brute-force algorithm is not good at all: if $|X| = k$ and $|\mathcal{S}| = n$, there are n^k substitutions $X \to \mathcal{S}$ whence the time spent by the algorithm is exponential of the size of the input.

Checking Identities

Given a semigroup $\mathcal{S}$, its identity checking problem, denoted $\textsc{Check-Id}(\mathcal{S})$, is a combinatorial decision problem whose instance is a semigroup identity $w \doteq w'$; the answer to the instance $w \doteq w'$ is "YES" if $\mathcal{S}$ satisfies $w \doteq w'$ and "NO" otherwise.

We stress that here $\mathcal{S}$ is fixed and it is the identity $w \doteq w'$ that serves as the input so that the time/space complexity should be measured in terms of the size of the identity, i.e., in terms of $|ww'|$.

For a finite semigroup, the identity checking problem is always decidable. Indeed, if $\mathcal{S}$ is finite, then for every identity $w \doteq w'$, there are only finitely many substitutions of elements in $\mathcal{S}$ for letters in $X := \mathrm{alph}(ww')$, and one can consecutively calculate the values of w and w' under each of these substitutions.

This brute-force algorithm is not good at all: if $|X| = k$ and $|\mathcal{S}| = n$, there are n^k substitutions $X \to \mathcal{S}$ whence the time spent by the algorithm is exponential of the size of the input.

Checking Identities

Given a semigroup $\mathcal{S}$, its identity checking problem, denoted $\textsc{Check-Id}(\mathcal{S})$, is a combinatorial decision problem whose instance is a semigroup identity $w \bumpeq w'$; the answer to the instance $w \bumpeq w'$ is "YES" if $\mathcal{S}$ satisfies $w \bumpeq w'$ and "NO" otherwise.

We stress that here $\mathcal{S}$ is fixed and it is the identity $w \bumpeq w'$ that serves as the input so that the time/space complexity should be measured in terms of the size of the identity, i.e., in terms of $|ww'|$.

For a finite semigroup, the identity checking problem is always decidable. Indeed, if $\mathcal{S}$ is finite, then for every identity $w \bumpeq w'$, there are only finitely many substitutions of elements in $\mathcal{S}$ for letters in $X := \mathrm{alph}(ww')$, and one can consecutively calculate the values of w and w' under each of these substitutions.

This brute-force algorithm is not good at all: if $|X| = k$ and $|\mathcal{S}| = n$, there are n^k substitutions $X \to \mathcal{S}$ whence the time spent by the algorithm is exponential of the size of the input.

Checking Identities

Given a semigroup $\mathcal{S}$, its identity checking problem, denoted $\textsc{Check-Id}(\mathcal{S})$, is a combinatorial decision problem whose instance is a semigroup identity $w \bumpeq w'$; the answer to the instance $w \bumpeq w'$ is "YES" if $\mathcal{S}$ satisfies $w \bumpeq w'$ and "NO" otherwise.

We stress that here $\mathcal{S}$ is fixed and it is the identity $w \bumpeq w'$ that serves as the input so that the time/space complexity should be measured in terms of the size of the identity, i.e., in terms of $|ww'|$.

For a finite semigroup, the identity checking problem is always decidable. Indeed, if $\mathcal{S}$ is finite, then for every identity $w \bumpeq w'$, there are only finitely many substitutions of elements in $\mathcal{S}$ for letters in $X := \mathrm{alph}(ww')$, and one can consecutively calculate the values of w and w' under each of these substitutions.

This brute-force algorithm is not good at all: if $|X| = k$ and $|\mathcal{S}| = n$, there are n^k substitutions $X \to \mathcal{S}$ whence the time spent by the algorithm is exponential of the size of the input.

Checking Identities

Given a semigroup $\mathcal{S}$, its identity checking problem, denoted $\textsc{Check-Id}(\mathcal{S})$, is a combinatorial decision problem whose instance is a semigroup identity $w \doteq w'$; the answer to the instance $w \doteq w'$ is "YES" if $\mathcal{S}$ satisfies $w \doteq w'$ and "NO" otherwise.

We stress that here $\mathcal{S}$ is fixed and it is the identity $w \doteq w'$ that serves as the input so that the time/space complexity should be measured in terms of the size of the identity, i.e., in terms of $|ww'|$.

For a finite semigroup, the identity checking problem is always decidable. Indeed, if $\mathcal{S}$ is finite, then for every identity $w \doteq w'$, there are only finitely many substitutions of elements in $\mathcal{S}$ for letters in $X := \mathrm{alph}(ww')$, and one can consecutively calculate the values of w and w' under each of these substitutions.

This brute-force algorithm is not good at all: if $|X| = k$ and $|\mathcal{S}| = n$, there are n^k substitutions $X \to \mathcal{S}$ whence the time spent by the algorithm is exponential of the size of the input.

Finite Case

On the other hand, for every finite semigroup $\mathcal{S}$, its identity checking problem belongs to the complexity class co-NP
(the class of negations of problems in NP).

Indeed, for every input $w \approx w'$ with $|\mathrm{alph}(ww')| = k$, one
1) guesses a k-tuple of elements in $\mathcal{S}$;
2) substitutes the elements from the guessed k-tuple
for the letters in $\mathrm{alph}(ww')$; and
3) checks if w and w' get different values under this substitution.
(The latter check can be performed since $\mathcal{S}$ is finite!)

Clearly, this nondeterministic algorithm has a chance to succeed iff $\mathcal{S}$ does not satisfy the identity $w \approx w'$.

The time spent by the algorithm is polynomial (in fact, linear) of the size of $w \approx w'$. Thus, $\textsc{Check-Id}(\mathcal{S})$ lies in co-NP.

Finite Case

On the other hand, for every finite semigroup $\mathcal{S}$, its identity checking problem belongs to the complexity class co-NP
(the class of negations of problems in NP).

Indeed, for every input $w \doteq w'$ with $|\mathsf{alph}(ww')| = k$, one
1) guesses a k-tuple of elements in $\mathcal{S}$;
2) substitutes the elements from the guessed k-tuple for the letters in $\mathsf{alph}(ww')$; and
3) checks if w and w' get different values under this substitution.
(The latter check can be performed since $\mathcal{S}$ is finite!)

Clearly, this nondeterministic algorithm has a chance to succeed iff $\mathcal{S}$ does not satisfy the identity $w \doteq w'$.

The time spent by the algorithm is polynomial (in fact, linear) of the size of $w \doteq w'$. Thus, $\textsc{Check-Id}(\mathcal{S})$ lies in co-NP.

Finite Case

On the other hand, for every finite semigroup $\mathcal{S}$, its identity checking problem belongs to the complexity class co-NP (the class of negations of problems in NP).

Indeed, for every input $w \doteq w'$ with $|\mathrm{alph}(ww')| = k$, one
1) guesses a k-tuple of elements in $\mathcal{S}$;
2) substitutes the elements from the guessed k-tuple for the letters in $\mathrm{alph}(ww')$; and
3) checks if w and w' get different values under this substitution. (The latter check can be performed since $\mathcal{S}$ is finite!)

Clearly, this nondeterministic algorithm has a chance to succeed iff $\mathcal{S}$ does not satisfy the identity $w \doteq w'$.

The time spent by the algorithm is polynomial (in fact, linear) of the size of $w \doteq w'$. Thus, CHECK-ID($\mathcal{S}$) lies in co-NP.

Finite Case

On the other hand, for every finite semigroup $\mathcal{S}$, its identity checking problem belongs to the complexity class co-NP
(the class of negations of problems in NP).

Indeed, for every input $w \doteq w'$ with $|\text{alph}(ww')| = k$, one
1) guesses a k-tuple of elements in $\mathcal{S}$;
2) substitutes the elements from the guessed k-tuple
for the letters in $\text{alph}(ww')$; and
3) checks if w and w' get different values under this substitution.
(The latter check can be performed since $\mathcal{S}$ is finite!)

Clearly, this nondeterministic algorithm has a chance to succeed iff $\mathcal{S}$ does not satisfy the identity $w \doteq w'$.

The time spent by the algorithm is polynomial (in fact, linear) of the size of $w \doteq w'$. Thus, CHECK-ID($\mathcal{S}$) lies in co-NP.

Finite Case

On the other hand, for every finite semigroup $\mathcal{S}$, its identity checking problem belongs to the complexity class co-NP
(the class of negations of problems in NP).

Indeed, for every input $w \doteq w'$ with $|\mathrm{alph}(ww')| = k$, one
1) guesses a k-tuple of elements in $\mathcal{S}$;
2) substitutes the elements from the guessed k-tuple
for the letters in $\mathrm{alph}(ww')$; and
3) checks if w and w' get different values under this substitution.
(The latter check can be performed since $\mathcal{S}$ is finite!)

Clearly, this nondeterministic algorithm has a chance to succeed iff $\mathcal{S}$ does not satisfy the identity $w \doteq w'$.

The time spent by the algorithm is polynomial (in fact, linear) of the size of $w \doteq w'$. Thus, $\textsc{Check-Id}(\mathcal{S})$ lies in co-NP.

Finite Case

On the other hand, for every finite semigroup $\mathcal{S}$, its identity checking problem belongs to the complexity class co-NP
(the class of negations of problems in NP).

Indeed, for every input $w \doteq w'$ with $|\mathrm{alph}(ww')| = k$, one
1) guesses a k-tuple of elements in $\mathcal{S}$;
2) substitutes the elements from the guessed k-tuple for the letters in $\mathrm{alph}(ww')$; and
3) checks if w and w' get different values under this substitution.
(The latter check can be performed since $\mathcal{S}$ is finite!)

Clearly, this nondeterministic algorithm has a chance to succeed iff $\mathcal{S}$ does not satisfy the identity $w \doteq w'$.

The time spent by the algorithm is polynomial (in fact, linear) of the size of $w \doteq w'$. Thus, $\textsc{Check-Id}(\mathcal{S})$ lies in co-NP.

Finite Case

On the other hand, for every finite semigroup $\mathcal{S}$, its identity checking problem belongs to the complexity class co-NP
(the class of negations of problems in NP).

Indeed, for every input $w \doteq w'$ with $|\mathrm{alph}(ww')| = k$, one
1) guesses a k-tuple of elements in $\mathcal{S}$;
2) substitutes the elements from the guessed k-tuple
for the letters in $\mathrm{alph}(ww')$; and
3) checks if w and w' get different values under this substitution.
(The latter check can be performed since $\mathcal{S}$ is finite!)

Clearly, this nondeterministic algorithm has a chance to succeed iff $\mathcal{S}$ does not satisfy the identity $w \doteq w'$.

The time spent by the algorithm is polynomial (in fact, linear) of the size of $w \doteq w'$. Thus, $\textsc{Check-Id}(\mathcal{S})$ lies in co-NP.

Finite Case

On the other hand, for every finite semigroup $\mathcal{S}$, its identity checking problem belongs to the complexity class co-NP
(the class of negations of problems in NP).

Indeed, for every input $w \doteq w'$ with $|\mathrm{alph}(ww')| = k$, one
1) guesses a k-tuple of elements in $\mathcal{S}$;
2) substitutes the elements from the guessed k-tuple
for the letters in $\mathrm{alph}(ww')$; and
3) checks if w and w' get different values under this substitution.
(The latter check can be performed since $\mathcal{S}$ is finite!)

Clearly, this nondeterministic algorithm has a chance to succeed iff $\mathcal{S}$ does not satisfy the identity $w \doteq w'$.

The time spent by the algorithm is polynomial (in fact, linear) of the size of $w \doteq w'$. Thus, $\textsc{Check-Id}(\mathcal{S})$ lies in co-NP.

Finite Case: Co-NP-completeness

Thus, we have an upper bound for the complexity of $\textsc{Check-Id}(\mathcal{S})$ with $\mathcal{S}$ being a finite semigroup.

This bound is tight: there are finite semigroups with co-NP-complete identity checking problem.

Examples: non-solvable groups (Gabor Horváth, John Lawrence, Laszlo Mérai, and Csaba Szabó, The complexity of the equivalence problem for nonsolvable groups, Bull. London Math. Soc. 39(3): 433–438 (2007)), the symmetric monoid on 3 points (Jorge Almeida, V., and Svetlana Goldberg, Complexity of the identity checking problem in finite semigroups, J. Math. Sci. 158(5): 605–614 (2009)), the 6-element Brandt monoid (Steve Seif, The Perkins semigroup has co-NP-complete term-equivalence problem, IJAC 15(2):317–326 (2005), and, independently, Ondřej Klíma, Complexity issues of checking identities in finite monoids, Semigroup Forum 79(3): 435–444 (2009)).

Finite Case: Co-NP-completeness

Thus, we have an upper bound for the complexity of $\textsc{Check-Id}(\mathcal{S})$ with $\mathcal{S}$ being a finite semigroup.

This bound is tight: there are finite semigroups with co-NP-complete identity checking problem.

Examples: non-solvable groups (Gabor Horváth, John Lawrence, Laszlo Mérai, and Csaba Szabó, The complexity of the equivalence problem for nonsolvable groups, Bull. London Math. Soc. 39(3): 433–438 (2007)), the symmetric monoid on 3 points (Jorge Almeida, V., and Svetlana Goldberg, Complexity of the identity checking problem in finite semigroups, J. Math. Sci. 158(5): 605–614 (2009)), the 6-element Brandt monoid (Steve Seif, The Perkins semigroup has co-NP-complete term-equivalence problem, IJAC 15(2):317–326 (2005), and, independently, Ondřej Klíma, Complexity issues of checking identities in finite monoids, Semigroup Forum 79(3): 435–444 (2009)).

Finite Case: Co-NP-completeness

Thus, we have an upper bound for the complexity of $\textsc{Check-Id}(\mathcal{S})$ with $\mathcal{S}$ being a finite semigroup.

This bound is tight: there are finite semigroups with co-NP-complete identity checking problem.

Examples: non-solvable groups (Gabor Horváth, John Lawrence, Laszlo Mérai, and Csaba Szabó, The complexity of the equivalence problem for nonsolvable groups, Bull. London Math. Soc. 39(3): 433–438 (2007)), the symmetric monoid on 3 points (Jorge Almeida, V., and Svetlana Goldberg, Complexity of the identity checking problem in finite semigroups, J. Math. Sci. 158(5): 605–614 (2009)), the 6-element Brandt monoid (Steve Seif, The Perkins semigroup has co-NP-complete term-equivalence problem, IJAC 15(2):317–326 (2005), and, independently, Ondřej Klíma, Complexity issues of checking identities in finite monoids, Semigroup Forum 79(3): 435–444 (2009)).

Finite Case: Co-NP-completeness

Thus, we have an upper bound for the complexity of $\textsc{Check-Id}(\mathcal{S})$ with $\mathcal{S}$ being a finite semigroup.

This bound is tight: there are finite semigroups with co-NP-complete identity checking problem.

Examples: non-solvable groups (Gabor Horváth, John Lawrence, Laszlo Mérai, and Csaba Szabó, The complexity of the equivalence problem for nonsolvable groups, Bull. London Math. Soc. 39(3): 433–438 (2007)), the symmetric monoid on 3 points (Jorge Almeida, V., and Svetlana Goldberg, Complexity of the identity checking problem in finite semigroups, J. Math. Sci. 158(5): 605–614 (2009)), the 6-element Brandt monoid (Steve Seif, The Perkins semigroup has co-NP-complete term-equivalence problem, IJAC 15(2):317–326 (2005), and, independently, Ondřej Klíma, Complexity issues of checking identities in finite monoids, Semigroup Forum 79(3): 435–444 (2009)).

Finite Case: Co-NP-completeness

Thus, we have an upper bound for the complexity of $\textsc{Check-Id}(\mathcal{S})$ with $\mathcal{S}$ being a finite semigroup.

This bound is tight: there are finite semigroups with co-NP-complete identity checking problem.

Examples: non-solvable groups (Gabor Horváth, John Lawrence, Laszlo Mérai, and Csaba Szabó, The complexity of the equivalence problem for nonsolvable groups, Bull. London Math. Soc. 39(3): 433–438 (2007)), the symmetric monoid on 3 points (Jorge Almeida, V., and Svetlana Goldberg, Complexity of the identity checking problem in finite semigroups, J. Math. Sci. 158(5): 605–614 (2009)), the 6-element Brandt monoid (Steve Seif, The Perkins semigroup has co-NP-complete term-equivalence problem, IJAC 15(2):317–326 (2005), and, independently, Ondřej Klíma, Complexity issues of checking identities in finite monoids, Semigroup Forum 79(3): 435–444 (2009)).

Finite Case: Open Problems

Main Problem

To classify finite semigroups $\mathcal{S}$ with respect to the complexity of $\textsc{Check-Id}(\mathcal{S})$: which semigroups are "easy" (the problem is in P) and which are "hard" (the problem is co-NP-complete)?

Open even in the group case (compare with rings).

For general finite semigroups, the behaviour of the complexity of $\textsc{Check-Id}(\mathcal{S})$ turns out to be very complex; e.g., an easy semigroup can contain a hard subsemigroup, etc.

Finite Case: Open Problems

Main Problem

To classify finite semigroups $\mathcal{S}$ with respect to the complexity of $\textsc{Check-Id}(\mathcal{S})$: which semigroups are "easy" (the problem is in P) and which are "hard" (the problem is co-NP-complete)?

Open even in the group case (compare with rings).

For general finite semigroups, the behaviour of the complexity of $\textsc{Check-Id}(\mathcal{S})$ turns out to be very complex; e.g., an easy semigroup can contain a hard subsemigroup, etc.

Finite Case: Open Problems

Main Problem

To classify finite semigroups $\mathcal{S}$ with respect to the complexity of $\textsc{Check-Id}(\mathcal{S})$: which semigroups are "easy" (the problem is in P) and which are "hard" (the problem is co-NP-complete)?

Open even in the group case (compare with rings).

For general finite semigroups, the behaviour of the complexity of $\textsc{Check-Id}(\mathcal{S})$ turns out to be very complex; e.g., an easy semigroup can contain a hard subsemigroup, etc.

Finite Case: Open Problems

Main Problem

To classify finite semigroups $\mathcal{S}$ with respect to the complexity of $\textsc{Check-Id}(\mathcal{S})$: which semigroups are "easy" (the problem is in P) and which are "hard" (the problem is co-NP-complete)?

Open even in the group case (compare with rings).

For general finite semigroups, the behaviour of the complexity of $\textsc{Check-Id}(\mathcal{S})$ turns out to be very complex; e.g., an easy semigroup can contain a hard subsemigroup, etc.

Dichotomy Problem

Is it true that every finite semigroup is either easy or hard?

Finite Case: Open Problems

Main Problem

To classify finite semigroups $\mathcal{S}$ with respect to the complexity of $\textsc{Check-Id}(\mathcal{S})$: which semigroups are "easy" (the problem is in P) and which are "hard" (the problem is co-NP-complete)?

Open even in the group case (compare with rings).

For general finite semigroups, the behaviour of the complexity of $\textsc{Check-Id}(\mathcal{S})$ turns out to be very complex; e.g., an easy semigroup can contain a hard subsemigroup, etc.

Dichotomy Problem

Is it true that every finite semigroup is either easy or hard?

Compare with CSP.

Identities of Infinite Semigroups

Identities in infinite semigroups are not well studied. Reason: "usual" infinite semigroups (transformations, relations, matrices) are too big (contain a copy of the non-monogenic free semigroup). Therefore they satisfy only trivial identities of the form $w \bumpeq w$.

But if an infinite semigroup does satisfy a non-trivial identity, its identity checking problem constitutes a challenge since no "finite" methods apply. Clearly, the brute-force approach fails as the number of k-tuples is infinite.

The nondeterministic guessing algorithm also fails in general because an infinite semigroup S may have undecidable word problem so that it might be impossible to decide whether or not the values of two words under a substitution are equal in S.

Vadim Murskiĭ (Examples of varieties of semigroups, Math. Notes 3(6): 423–427 (1968)) constructed an infinite semigroup $\mathcal{M}$ such that the problem $\textsc{Check-Id}(\mathcal{M})$ is undecidable.

Identities of Infinite Semigroups

Identities in infinite semigroups are not well studied. Reason: "usual" infinite semigroups (transformations, relations, matrices) are too big (contain a copy of the non-monogenic free semigroup). Therefore they satisfy only trivial identities of the form $w \doteq w$.

But if an infinite semigroup does satisfy a non-trivial identity, its identity checking problem constitutes a challenge since no "finite" methods apply. Clearly, the brute-force approach fails as the number of k-tuples is infinite.

The nondeterministic guessing algorithm also fails in general because an infinite semigroup $\mathcal{S}$ may have undecidable word problem so that it might be impossible to decide whether or not the values of two words under a substitution are equal in $\mathcal{S}$.

Vadim Murskiĭ (Examples of varieties of semigroups, Math. Notes 3(6): 423–427 (1968)) constructed an infinite semigroup $\mathcal{M}$ such that the problem $\textsc{Check-Id}(\mathcal{M})$ is undecidable.

Identities of Infinite Semigroups

Identities in infinite semigroups are not well studied. Reason: "usual" infinite semigroups (transformations, relations, matrices) are too big (contain a copy of the non-monogenic free semigroup). Therefore they satisfy only trivial identities of the form $w \doteq w$.

But if an infinite semigroup does satisfy a non-trivial identity, its identity checking problem constitutes a challenge since no "finite" methods apply. Clearly, the brute-force approach fails as the number of k-tuples is infinite.

The nondeterministic guessing algorithm also fails in general because an infinite semigroup $\mathcal{S}$ may have undecidable word problem so that it might be impossible to decide whether or not the values of two words under a substitution are equal in $\mathcal{S}$.

Vadim Murskiĭ (Examples of varieties of semigroups, Math. Notes 3(6): 423–427 (1968)) constructed an infinite semigroup $\mathcal{M}$ such that the problem $\textsc{Check-Id}(\mathcal{M})$ is undecidable.

Identities of Infinite Semigroups

Identities in infinite semigroups are not well studied. Reason: "usual" infinite semigroups (transformations, relations, matrices) are too big (contain a copy of the non-monogenic free semigroup). Therefore they satisfy only trivial identities of the form $w \bumpeq w$.

But if an infinite semigroup does satisfy a non-trivial identity, its identity checking problem constitutes a challenge since no "finite" methods apply. Clearly, the brute-force approach fails as the number of k-tuples is infinite.

The nondeterministic guessing algorithm also fails in general because an infinite semigroup $\mathcal{S}$ may have undecidable word problem so that it might be impossible to decide whether or not the values of two words under a substitution are equal in $\mathcal{S}$.

Vadim Murskiĭ (Examples of varieties of semigroups, Math. Notes 3(6): 423–427 (1968)) constructed an infinite semigroup $\mathcal{M}$ such that the problem $\textsc{Check-Id}(\mathcal{M})$ is undecidable.

Identities of Infinite Semigroups

Identities in infinite semigroups are not well studied. Reason: "usual" infinite semigroups (transformations, relations, matrices) are too big (contain a copy of the non-monogenic free semigroup). Therefore they satisfy only trivial identities of the form $w \bumpeq w$.

But if an infinite semigroup does satisfy a non-trivial identity, its identity checking problem constitutes a challenge since no "finite" methods apply. Clearly, the brute-force approach fails as the number of k-tuples is infinite.

The nondeterministic guessing algorithm also fails in general because an infinite semigroup $\mathcal{S}$ may have undecidable word problem so that it might be impossible to decide whether or not the values of two words under a substitution are equal in $\mathcal{S}$.

Vadim Murskiĭ (Examples of varieties of semigroups, Math. Notes 3(6): 423–427 (1968)) constructed an infinite semigroup $\mathcal{M}$ such that the problem $\textsc{Check-Id}(\mathcal{M})$ is undecidable.

Identities of Infinite Semigroups

Identities in infinite semigroups are not well studied. Reason: "usual" infinite semigroups (transformations, relations, matrices) are too big (contain a copy of the non-monogenic free semigroup). Therefore they satisfy only trivial identities of the form $w \bumpeq w$.

But if an infinite semigroup does satisfy a non-trivial identity, its identity checking problem constitutes a challenge since no "finite" methods apply. Clearly, the brute-force approach fails as the number of k-tuples is infinite.

The nondeterministic guessing algorithm also fails in general because an infinite semigroup $\mathcal{S}$ may have undecidable word problem so that it might be impossible to decide whether or not the values of two words under a substitution are equal in $\mathcal{S}$.

Vadim Murskiĭ (Examples of varieties of semigroups, Math. Notes 3(6): 423–427 (1968)) constructed an infinite semigroup $\mathcal{M}$ such that the problem $\textsc{Check-Id}(\mathcal{M})$ is undecidable.

Identities of Infinite Semigroups

Identities in infinite semigroups are not well studied. Reason: "usual" infinite semigroups (transformations, relations, matrices) are too big (contain a copy of the non-monogenic free semigroup). Therefore they satisfy only trivial identities of the form $w \bumpeq w$.

But if an infinite semigroup does satisfy a non-trivial identity, its identity checking problem constitutes a challenge since no "finite" methods apply. Clearly, the brute-force approach fails as the number of k-tuples is infinite.

The nondeterministic guessing algorithm also fails in general because an infinite semigroup $\mathcal{S}$ may have undecidable word problem so that it might be impossible to decide whether or not the values of two words under a substitution are equal in $\mathcal{S}$.

Vadim Murskiĭ (Examples of varieties of semigroups, Math. Notes 3(6): 423–427 (1968)) constructed an infinite semigroup $\mathcal{M}$ such that the problem $\textsc{Check-Id}(\mathcal{M})$ is undecidable.

Identities of Infinite Semigroups: Approach

How can one recognize the identities of an infinite semigroup $\mathcal{S}$ when all "finite" methods fail? One should look for a combinatorial characterization of the identities of $\mathcal{S}$ that one could effectively verify for each input $w \doteq w'$ instead of evaluating w and w' in $\mathcal{S}$.

Toy example: the Parikh vector of a word w is

$$p(w) := (|w|_{a_1}, |w|_{a_2}, \dots, |w|_{a_k}),$$

where $\mathrm{alph}(w) = \{a_1, \dots, a_n\}$ and $|w|_{a_i}$ denotes the number of occurrences of the letter a_i in the word w. For instance, $p(xy^2zxzy^2x) = (3, 4, 2)$.

An identity $w \doteq w'$ holds in the additive (or multiplicative) semigroup $\mathbb{N}$ iff $p(w) = p(w')$. Hence, $\textsc{Check-Id}(\mathbb{N})$ is in P.

Identities of Infinite Semigroups: Approach

How can one recognize the identities of an infinite semigroup $\mathcal{S}$ when all "finite" methods fail? One should look for a combinatorial characterization of the identities of $\mathcal{S}$ that one could effectively verify for each input $w \bumpeq w'$ instead of evaluating w and w' in $\mathcal{S}$.

Toy example: the Parikh vector of a word w is

$$p(w) := (|w|_{a_1}, |w|_{a_2}, \dots, |w|_{a_k}),$$

where $\mathrm{alph}(w) = \{a_1, \dots, a_n\}$ and $|w|_{a_i}$ denotes the number of occurrences of the letter a_i in the word w. For instance, $p(xy^2zxzy^2x) = (3, 4, 2)$.

An identity $w \bumpeq w'$ holds in the additive (or multiplicative) semigroup $\mathbb{N}$ iff $p(w) = p(w')$. Hence, $\textsc{Check-Id}(\mathbb{N})$ is in P.

Identities of Infinite Semigroups: Approach

How can one recognize the identities of an infinite semigroup $\mathcal{S}$ when all "finite" methods fail? One should look for a combinatorial characterization of the identities of $\mathcal{S}$ that one could effectively verify for each input $w \bumpeq w'$ instead of evaluating w and w' in $\mathcal{S}$.

Toy example: the Parikh vector of a word w is

$$p(w) := (|w|_{a_1}, |w|_{a_2}, \dots, |w|_{a_k}),$$

where $\mathrm{alph}(w) = \{a_1, \dots, a_n\}$ and $|w|_{a_i}$ denotes the number of occurrences of the letter a_i in the word w. For instance, $p(xy^2zxzy^2x) = (3, 4, 2)$.

An identity $w \bumpeq w'$ holds in the additive (or multiplicative) semigroup $\mathbb{N}$ iff $p(w) = p(w')$. Hence, $\textsc{Check-Id}(\mathbb{N})$ is in P.

Identities of Infinite Semigroups: Approach

How can one recognize the identities of an infinite semigroup $\mathcal{S}$ when all "finite" methods fail? One should look for a combinatorial characterization of the identities of $\mathcal{S}$ that one could effectively verify for each input $w \doteq w'$ instead of evaluating w and w' in $\mathcal{S}$.

Toy example: the Parikh vector of a word w is

$$p(w) := (|w|_{a_1}, |w|_{a_2}, \ldots, |w|_{a_k}),$$

where $\mathrm{alph}(w) = \{a_1, \ldots, a_n\}$ and $|w|_{a_i}$ denotes the number of occurrences of the letter a_i in the word w. For instance, $p(xy^2zxzy^2x) = (3, 4, 2)$.

An identity $w \doteq w'$ holds in the additive (or multiplicative) semigroup $\mathbb{N}$ iff $p(w) = p(w')$. Hence, $\textsc{Check-Id}(\mathbb{N})$ is in P.

Identities of Infinite Semigroups: Approach

How can one recognize the identities of an infinite semigroup $\mathcal{S}$ when all "finite" methods fail? One should look for a combinatorial characterization of the identities of $\mathcal{S}$ that one could effectively verify for each input $w \doteq w'$ instead of evaluating w and w' in $\mathcal{S}$.

Toy example: the Parikh vector of a word w is

$$p(w) := (|w|_{a_1}, |w|_{a_2}, \dots, |w|_{a_k}),$$

where $\mathrm{alph}(w) = \{a_1, \dots, a_n\}$ and $|w|_{a_i}$ denotes the number of occurrences of the letter a_i in the word w. For instance, $p(xy^2zxzy^2x) = (3, 4, 2)$.

An identity $w \doteq w'$ holds in the additive (or multiplicative) semigroup $\mathbb{N}$ iff $p(w) = p(w')$. Hence, CHECK-ID($\mathbb{N}$) is in P.

Identities of Infinite Semigroups: Approach

How can one recognize the identities of an infinite semigroup $\mathcal{S}$ when all "finite" methods fail? One should look for a combinatorial characterization of the identities of $\mathcal{S}$ that one could effectively verify for each input $w \doteq w'$ instead of evaluating w and w' in $\mathcal{S}$.

Toy example: the Parikh vector of a word w is

$$p(w) := (|w|_{a_1}, |w|_{a_2}, \dots, |w|_{a_k}),$$

where $\mathrm{alph}(w) = \{a_1, \dots, a_n\}$ and $|w|_{a_i}$ denotes the number of occurrences of the letter a_i in the word w. For instance, $p(xy^2zxzy^2x) = (3, 4, 2)$.

An identity $w \doteq w'$ holds in the additive (or multiplicative) semigroup $\mathbb{N}$ iff $p(w) = p(w')$. Hence, $\textsc{Check-Id}(\mathbb{N})$ is in P.

Identities of Infinite Semigroups: Examples

Nontrivial facts about $\textsc{Check-Id}(\mathcal{S})$ with $\mathcal{S}$ infinite are sparse.

Nontrivial facts about $\textsc{Check-Id}(\mathcal{S})$ with $\mathcal{S}$ infinite are sparse.

Theorem (Lev Shneerson and V., The identities of the free product of two trivial semigroups, Semigroup Forum 95(1): 245–250 (2017))

Let $\mathcal{J}_\infty := \langle e, f \mid e^2 = e,\ f^2 = f\rangle$. $\textsc{Check-Id}(\mathcal{J}_\infty)$ is in P.

Identities of Infinite Semigroups: Examples

Nontrivial facts about $\textsc{Check-Id}(\mathcal{S})$ with $\mathcal{S}$ infinite are sparse.

Theorem (Lev Shneerson and V., The identities of the free product of two trivial semigroups, Semigroup Forum 95(1): 245–250 (2017))

Let $\mathcal{J}_\infty := \langle e, f \mid e^2 = e,\ f^2 = f\rangle$. $\textsc{Check-Id}(\mathcal{J}_\infty)$ is in P.

The bicyclic monoid $\mathcal{B} := \langle b, c \mid cb = 1\rangle$ plays a distinguished role in semigroup theory.

Identities of Infinite Semigroups: Examples

Nontrivial facts about $\textsc{Check-Id}(\mathcal{S})$ with $\mathcal{S}$ infinite are sparse.

Theorem (Lev Shneerson and V., The identities of the free product of two trivial semigroups, Semigroup Forum 95(1): 245–250 (2017))

Let $\mathcal{J}_\infty := \langle e, f \mid e^2 = e,\ f^2 = f\rangle$. $\textsc{Check-Id}(\mathcal{J}_\infty)$ is in P.

The bicyclic monoid $\mathcal{B} := \langle b, c \mid cb = 1\rangle$ plays a distinguished role in semigroup theory. Sergey Adian (Identities in special semigroups, Soviet Math. Dokl. 3: 401–404 (1962)) discovered that $\mathcal{B}$ satisfies the identity $xy^2xyx^2y^2x \bumpeq xy^2x^2yxy^2x$.

Identities of Infinite Semigroups: Examples

Nontrivial facts about $\textsc{Check-Id}(\mathcal{S})$ with $\mathcal{S}$ infinite are sparse.

Theorem (Lev Shneerson and V., The identities of the free product of two trivial semigroups, Semigroup Forum 95(1): 245–250 (2017))

Let $\mathcal{J}_\infty := \langle e, f \mid e^2 = e,\ f^2 = f \rangle$. $\textsc{Check-Id}(\mathcal{J}_\infty)$ is in P.

The bicyclic monoid $\mathcal{B} := \langle b, c \mid cb = 1 \rangle$ plays a distinguished role in semigroup theory. Sergey Adian (Identities in special semigroups, Soviet Math. Dokl. 3: 401–404 (1962)) discovered that $\mathcal{B}$ satisfies the identity $xy^2xyx^2y^2x \bumpeq xy^2x^2yxy^2x$.

Theorem (Laura Daviaud, Marianne Johnson, and Mark Kambites, Identities in upper triangular tropical matrix semigroups and the bicyclic monoid, J. Algebra 501: 503–525 (2018))

$\textsc{Check-Id}(\mathcal{B})$ is in P.

Temperley–Lieb Algebras

Neville Temperley and Elliott Lieb (Relations between the 'percolation' and 'colouring' problem and other graph-theoretical problems associated with regular planar lattices: Some exact results for the percolation problem, Proc. Roy. Soc. London Ser. A 322, 251–280, 1971) motivated by some problems in statistical physics have introduced what is now called Temperley–Lieb algebras. These are associative linear algebras with 1 over a commutative ring R.

Given n and $\delta \in R$, the algebra $\mathcal{TL}_n(\delta)$ is generated by $n-1$ generators $h_1, \dots, h_{n-1}$ subject to the relations

$$\begin{aligned} h_i h_j &= h_j h_i && \text{if } |i-j| \geq 2, \\ h_i h_j h_i &= h_i && \text{if } |i-j| = 1, \\ h_i h_i &= \delta h_i. \end{aligned}$$

Temperley–Lieb Algebras

Neville Temperley and Elliott Lieb (Relations between the 'percolation' and 'colouring' problem and other graph-theoretical problems associated with regular planar lattices: Some exact results for the percolation problem, Proc. Roy. Soc. London Ser. A 322, 251–280, 1971) motivated by some problems in statistical physics have introduced what is now called Temperley–Lieb algebras. These are associative linear algebras with 1 over a commutative ring R.

Given n and $\delta \in R$, the algebra $\mathcal{TL}_n(\delta)$ is generated by $n-1$ generators $h_1, \ldots, h_{n-1}$ subject to the relations

$$\begin{aligned} h_i h_j &= h_j h_i \quad \text{if } |i-j| \geq 2, \\ h_i h_j h_i &= h_i \quad \text{if } |i-j| = 1, \\ h_i h_i &= \delta h_i. \end{aligned}$$

Temperley–Lieb Algebras

Neville Temperley and Elliott Lieb (Relations between the 'percolation' and 'colouring' problem and other graph-theoretical problems associated with regular planar lattices: Some exact results for the percolation problem, Proc. Roy. Soc. London Ser. A 322, 251–280, 1971) motivated by some problems in statistical physics have introduced what is now called Temperley–Lieb algebras. These are associative linear algebras with 1 over a commutative ring R.

Given n and $\delta \in R$, the algebra $\mathcal{TL}_n(\delta)$ is generated by $n-1$ generators $h_1, \dots, h_{n-1}$ subject to the relations

$$\begin{aligned} h_i h_j &= h_j h_i \quad \text{if } |i-j| \geq 2, \\ h_i h_j h_i &= h_i \quad \text{if } |i-j| = 1, \\ h_i h_i &= \delta h_i. \end{aligned}$$

Kauffman Monoids

The relations of $\mathcal{TL}_n(\delta)$ do not involve addition. This suggests introducing a monoid whose monoid algebra over R could be identified with $\mathcal{TL}_n(\delta)$. A tiny obstacle is the scalar δ in $h_ih_i = \delta h_i$. It can be bypassed by adding a new generator c that imitates δ.

This way one arrives at the monoid $\mathcal{K}_n$ with n generators $c, h_1, \dots, h_{n-1}$ subject to the relations

$$
\begin{aligned}
&h_ih_j = h_jh_i \quad \text{if } |i-j| \geq 2,\\
&h_ih_jh_i = h_i \quad \text{if } |i-j| = 1,\\
&h_ih_i = ch_i = h_ic.
\end{aligned}
$$

The monoids $\mathcal{K}_n$ are called the Kauffman monoids. Lois Kauffman (An invariant of regular isotopy, Trans. Amer. Math. Soc. 318: 417–471 (1990)) independently invented these monoids as geometrical objects. We present Kauffman's approach in a slightly more general setting.

Kauffman Monoids

The relations of $\mathcal{TL}_n(\delta)$ do not involve addition. This suggests introducing a monoid whose monoid algebra over R could be identified with $\mathcal{TL}_n(\delta)$.

A tiny obstacle is the scalar δ in $h_ih_i = \delta h_i$. It can be bypassed by adding a new generator c that imitates δ.

This way one arrives at the monoid $\mathcal{K}_n$ with n generators $c, h_1, \dots, h_{n-1}$ subject to the relations

$$\begin{aligned} h_ih_j &= h_jh_i \quad \text{if } |i-j| \geq 2, \\ h_ih_jh_i &= h_i \quad \text{if } |i-j| = 1, \\ h_ih_i &= ch_i = h_ic. \end{aligned}$$

The monoids $\mathcal{K}_n$ are called the Kauffman monoids. Lois Kauffman (An invariant of regular isotopy, Trans. Amer. Math. Soc. 318: 417–471 (1990)) independently invented these monoids as geometrical objects. We present Kauffman's approach in a slightly more general setting.

Kauffman Monoids

The relations of $\mathcal{TL}_n(\delta)$ do not involve addition. This suggests introducing a monoid whose monoid algebra over R could be identified with $\mathcal{TL}_n(\delta)$. A tiny obstacle is the scalar δ in $h_i h_i = \delta h_i$. It can be bypassed by adding a new generator c that imitates δ.

This way one arrives at the monoid $\mathcal{K}_n$ with n generators $c, h_1, \dots, h_{n-1}$ subject to the relations

$$\begin{aligned} h_i h_j &= h_j h_i \quad \text{if } |i-j| \geq 2, \\ h_i h_j h_i &= h_i \quad \text{if } |i-j| = 1, \\ h_i h_i &= c h_i = h_i c. \end{aligned}$$

The monoids $\mathcal{K}_n$ are called the Kauffman monoids. Lois Kauffman (An invariant of regular isotopy, Trans. Amer. Math. Soc. 318: 417–471 (1990)) independently invented these monoids as geometrical objects. We present Kauffman's approach in a slightly more general setting.

Kauffman Monoids

The relations of $\mathcal{TL}_n(\delta)$ do not involve addition. This suggests introducing a monoid whose monoid algebra over R could be identified with $\mathcal{TL}_n(\delta)$. A tiny obstacle is the scalar δ in $h_ih_i = \delta h_i$. It can be bypassed by adding a new generator c that imitates δ.

This way one arrives at the monoid $\mathcal{K}_n$ with n generators $c, h_1, \dots, h_{n-1}$ subject to the relations

$$\begin{aligned} h_ih_j &= h_jh_i \quad \text{if } |i-j| \geq 2, \\ h_ih_jh_i &= h_i \quad \text{if } |i-j| = 1, \\ h_ih_i &= ch_i = h_ic. \end{aligned}$$

The monoids $\mathcal{K}_n$ are called the Kauffman monoids. Lois Kauffman (An invariant of regular isotopy, Trans. Amer. Math. Soc. 318: 417–471 (1990)) independently invented these monoids as geometrical objects. We present Kauffman's approach in a slightly more general setting.

Kauffman Monoids

The relations of $\mathcal{TL}_n(\delta)$ do not involve addition. This suggests introducing a monoid whose monoid algebra over R could be identified with $\mathcal{TL}_n(\delta)$. A tiny obstacle is the scalar δ in $h_ih_i = \delta h_i$. It can be bypassed by adding a new generator c that imitates δ.

This way one arrives at the monoid $\mathcal{K}_n$ with n generators $c, h_1, \dots, h_{n-1}$ subject to the relations

$$\begin{aligned} h_ih_j &= h_jh_i \quad \text{if } |i-j| \geq 2, \\ h_ih_jh_i &= h_i \quad \text{if } |i-j| = 1, \\ h_ih_i &= ch_i = h_ic. \end{aligned}$$

The monoids $\mathcal{K}_n$ are called the Kauffman monoids. Lois Kauffman (An invariant of regular isotopy, Trans. Amer. Math. Soc. 318: 417–471 (1990)) independently invented these monoids as geometrical objects. We present Kauffman's approach in a slightly more general setting.

Kauffman Monoids

The relations of $\mathcal{TL}_n(\delta)$ do not involve addition. This suggests introducing a monoid whose monoid algebra over R could be identified with $\mathcal{TL}_n(\delta)$. A tiny obstacle is the scalar δ in $h_ih_i = \delta h_i$. It can be bypassed by adding a new generator c that imitates δ.

This way one arrives at the monoid $\mathcal{K}_n$ with n generators $c, h_1, \dots, h_{n-1}$ subject to the relations

$$\begin{aligned} h_ih_j &= h_jh_i \quad \text{if } |i-j| \geq 2,\\ h_ih_jh_i &= h_i \quad \text{if } |i-j| = 1,\\ h_ih_i &= ch_i = h_ic. \end{aligned}$$

The monoids $\mathcal{K}_n$ are called the Kauffman monoids. Lois Kauffman (An invariant of regular isotopy, Trans. Amer. Math. Soc. 318: 417–471 (1990)) independently invented these monoids as geometrical objects. We present Kauffman's approach in a slightly more general setting.

Kauffman Monoids

The relations of $\mathcal{TL}_n(\delta)$ do not involve addition. This suggests introducing a monoid whose monoid algebra over R could be identified with $\mathcal{TL}_n(\delta)$. A tiny obstacle is the scalar δ in $h_ih_i = \delta h_i$. It can be bypassed by adding a new generator c that imitates δ.

This way one arrives at the monoid $\mathcal{K}_n$ with n generators $c, h_1, \dots, h_{n-1}$ subject to the relations

$$\begin{aligned}
h_ih_j &= h_jh_i \quad \text{if } |i-j| \geq 2,\\
h_ih_jh_i &= h_i \quad \text{if } |i-j| = 1,\\
h_ih_i &= ch_i = h_ic.
\end{aligned}$$

The monoids $\mathcal{K}_n$ are called the Kauffman monoids. Lois Kauffman (An invariant of regular isotopy, Trans. Amer. Math. Soc. 318: 417–471 (1990)) independently invented these monoids as geometrical objects. We present Kauffman's approach in a slightly more general setting.

Kauffman Monoids

The relations of $\mathcal{TL}_n(\delta)$ do not involve addition. This suggests introducing a monoid whose monoid algebra over R could be identified with $\mathcal{TL}_n(\delta)$. A tiny obstacle is the scalar δ in $h_ih_i = \delta h_i$. It can be bypassed by adding a new generator c that imitates δ.

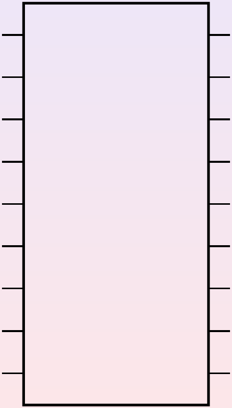
This way one arrives at the monoid $\mathcal{K}_n$ with n generators $c, h_1, \dots, h_{n-1}$ subject to the relations

$$\begin{aligned} h_ih_j &= h_jh_i \quad \text{if } |i-j| \geq 2, \\ h_ih_jh_i &= h_i \quad \text{if } |i-j| = 1, \\ h_ih_i &= ch_i = h_ic. \end{aligned}$$

The monoids $\mathcal{K}_n$ are called the Kauffman monoids. Lois Kauffman (An invariant of regular isotopy, Trans. Amer. Math. Soc. 318: 417–471 (1990)) independently invented these monoids as geometrical objects. We present Kauffman's approach in a slightly more general setting.

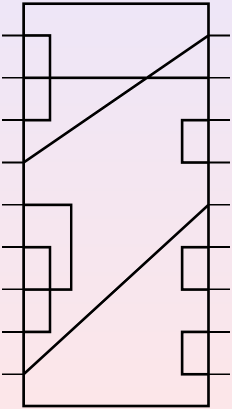
Wire Monoids

Fix n and consider "chips" with $2n$ pins, n on each side. Pins are connected by n wires. To multiply chips, we connect the right hand side pins of the first with the left hand side pins of the second.



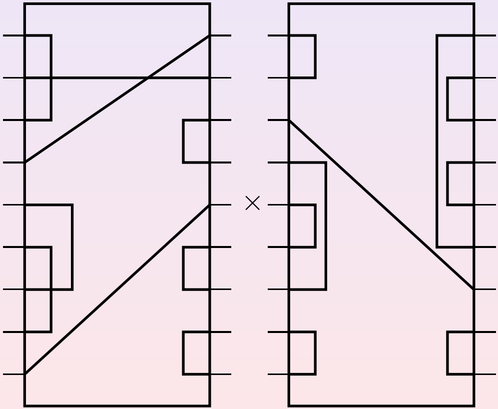
Wire Monoids

Fix n and consider "chips" with $2n$ pins, n on each side. Pins are connected by n wires. To multiply chips, we connect the right hand side pins of the first with the left hand side pins of the second.



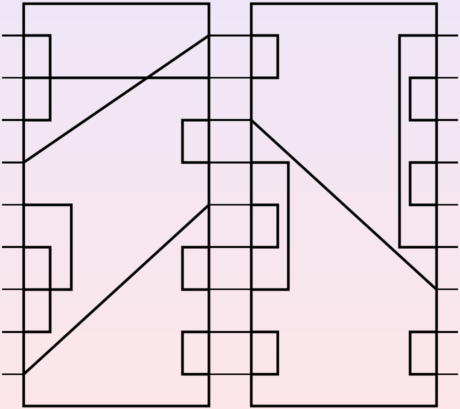
Wire Monoids

Fix n and consider "chips" with $2n$ pins, n on each side. Pins are connected by n wires. To multiply chips, we connect the right hand side pins of the first with the left hand side pins of the second.



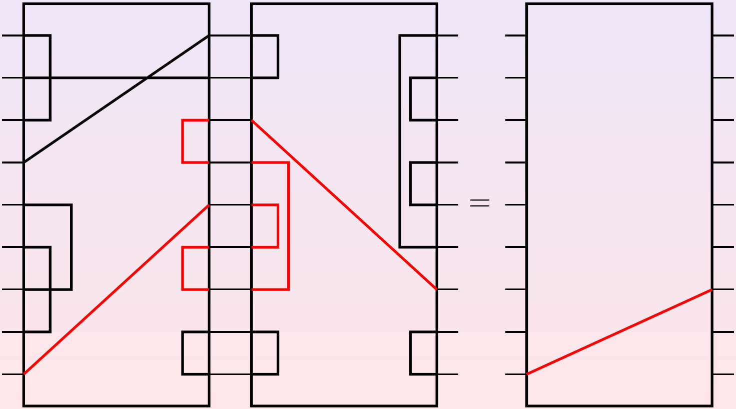
Wire Monoids

Fix n and consider "chips" with $2n$ pins, n on each side. Pins are connected by n wires. To multiply chips, we connect the right hand side pins of the first with the left hand side pins of the second.



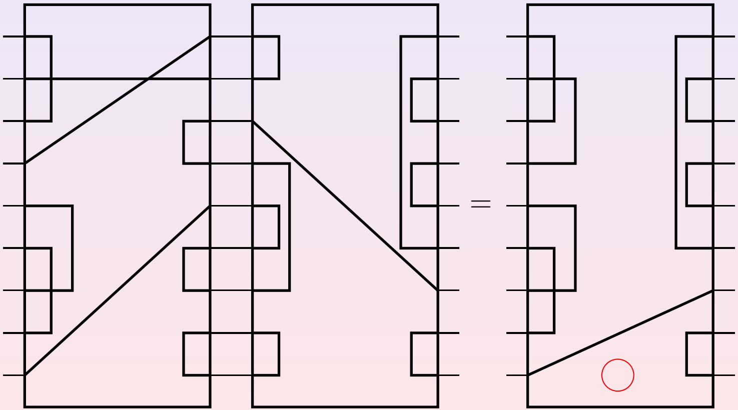
Wire Monoids

Fix n and consider "chips" with $2n$ pins, n on each side. Pins are connected by n wires. To multiply chips, we connect the right hand side pins of the first with the left hand side pins of the second.



Wire Monoids

Fix n and consider "chips" with $2n$ pins, n on each side. Pins are connected by n wires. To multiply chips, we connect the right hand side pins of the first with the left hand side pins of the second.



Types of Wire Monoids

There are two issues on which the outcome of the above definition depends: 1) do we care of crossing wires? 2) do we care of circles?

	Ignore circles	Count circles
Crossings OK	Brauer monoids	Wire monoids
No crossings	Jones monoids	Kauffman monoids

Types of Wire Monoids

There are two issues on which the outcome of the above definition depends: 1) do we care of crossing wires? 2) do we care of circles?

	Ignore circles	Count circles
Crossings OK	Brauer monoids	Wire monoids
No crossings	Jones monoids	Kauffman monoids

Types of Wire Monoids

There are two issues on which the outcome of the above definition depends: 1) do we care of crossing wires? 2) do we care of circles?

	Ignore circles	Count circles
Crossings OK	Brauer monoids	Wire monoids
No crossings	Jones monoids	Kauffman monoids

Types of Wire Monoids

There are two issues on which the outcome of the above definition depends: 1) do we care of crossing wires? 2) do we care of circles?

	Ignore circles	Count circles
Crossings OK	Brauer monoids	Wire monoids
No crossings	Jones monoids	Kauffman monoids

Richard Brauer's monoids arose in his paper: On algebras which are connected with the semisimple continuous groups, Ann. Math. 38: 857–872 (1937), as vector space bases of certain associative algebras relevant in representation theory.

Types of Wire Monoids

There are two issues on which the outcome of the above definition depends: 1) do we care of crossing wires? 2) do we care of circles?

	Ignore circles	Count circles
Crossings OK	Brauer monoids	Wire monoids
No crossings	Jones monoids	Kauffman monoids

Jones monoids are named after Vaughan Jones, the famous knot theorist. They are sometimes called Temperley–Lieb monoids.

Types of Wire Monoids

There are two issues on which the outcome of the above definition depends: 1) do we care of crossing wires? 2) do we care of circles?

	Ignore circles	Count circles
Crossings OK	Brauer monoids	Wire monoids
No crossings	Jones monoids	Kauffman monoids

Types of Wire Monoids

There are two issues on which the outcome of the above definition depends: 1) do we care of crossing wires? 2) do we care of circles?

	Ignore circles	Count circles
Crossings OK	Brauer monoids	Wire monoids
No crossings	Jones monoids	Kauffman monoids

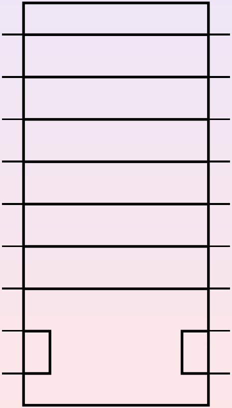
Kauffman monoids arise when crossing are not allowed, and we care of the number of circles.

Kauffman Monoids as Wire Monoids

Thus, the Kauffman monoid $\mathcal{K}_n$ consists of $2n$-pin chips with non-crossing wires that may contain circles. Only the number of circles matters, not their location. The monoid $\mathcal{K}_n$ is generated by the hooks $h_1, \ldots, h_{n-1}$ and the circle c.

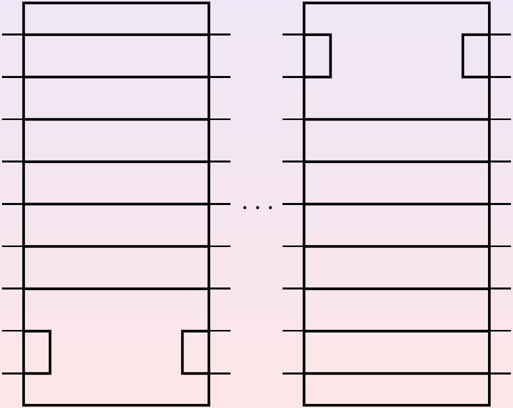
Kauffman Monoids as Wire Monoids

Thus, the Kauffman monoid $\mathcal{K}_n$ consists of $2n$-pin chips with non-crossing wires that may contain circles. Only the number of circles matters, not their location. The monoid $\mathcal{K}_n$ is generated by the hooks $h_1, \ldots, h_{n-1}$ and the circle c.



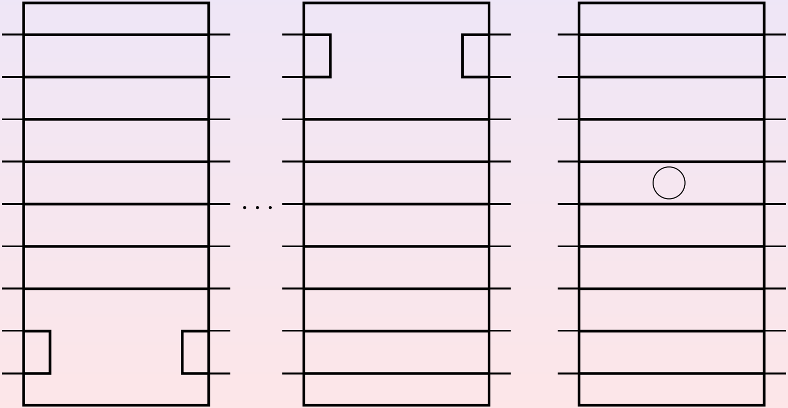
Kauffman Monoids as Wire Monoids

Thus, the Kauffman monoid $\mathcal{K}_n$ consists of $2n$-pin chips with non-crossing wires that may contain circles. Only the number of circles matters, not their location. The monoid $\mathcal{K}_n$ is generated by the hooks $h_1, \dots, h_{n-1}$ and the circle c.



Kauffman Monoids as Wire Monoids

Thus, the Kauffman monoid $\mathcal{K}_n$ consists of $2n$-pin chips with non-crossing wires that may contain circles. Only the number of circles matters, not their location. The monoid $\mathcal{K}_n$ is generated by the hooks $h_1, \ldots, h_{n-1}$ and the circle c.



Kauffman Monoids as Wire Monoids, continued

Recall the relations we used to define $\mathcal{K}_n$:

$$
\begin{aligned}
h_i h_j &= h_j h_i \quad \text{if } |i-j| \geq 2,\\
h_i h_j h_i &= h_i \quad \text{if } |i-j| = 1,\\
h_i h_i &= c h_i,\\
c h_i &= h_i c.
\end{aligned}
$$

These relations are satisfied when h_i and c are interpreted as the hooks and the circle. For the last relation it is clear— the circle does not react with the hooks, for the others it is shown in the next slides.

Kauffman Monoids as Wire Monoids, continued

Recall the relations we used to define $\mathcal{K}_n$:

$$\begin{aligned} h_i h_j &= h_j h_i \quad \text{if } |i-j| \geq 2, \\ h_i h_j h_i &= h_i \quad \text{if } |i-j| = 1, \\ h_i h_i &= c h_i, \\ c h_i &= h_i c. \end{aligned}$$

These relations are satisfied when h_i and c are interpreted as the hooks and the circle. For the last relation it is clear— the circle does not react with the hooks, for the others it is shown in the next slides.

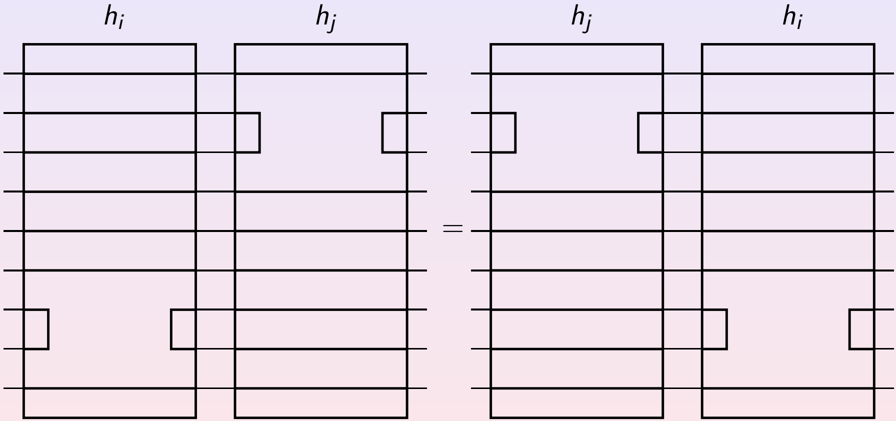
Kauffman Monoids as Wire Monoids, continued

Recall the relations we used to define $\mathcal{K}_n$:

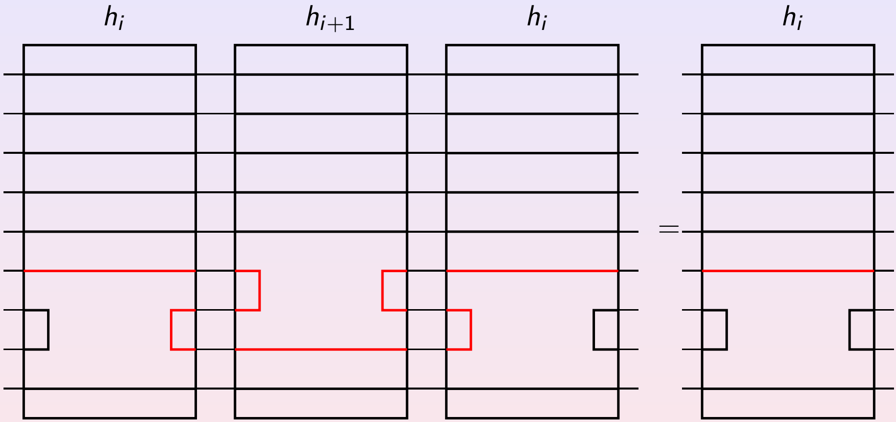
$$\begin{aligned} h_i h_j &= h_j h_i \quad \text{if } |i-j| \geq 2, \\ h_i h_j h_i &= h_i \quad \text{if } |i-j| = 1, \\ h_i h_i &= c h_i, \\ c h_i &= h_i c. \end{aligned}$$

These relations are satisfied when h_i and c are interpreted as the hooks and the circle. For the last relation it is clear— the circle does not react with the hooks, for the others it is shown in the next slides.

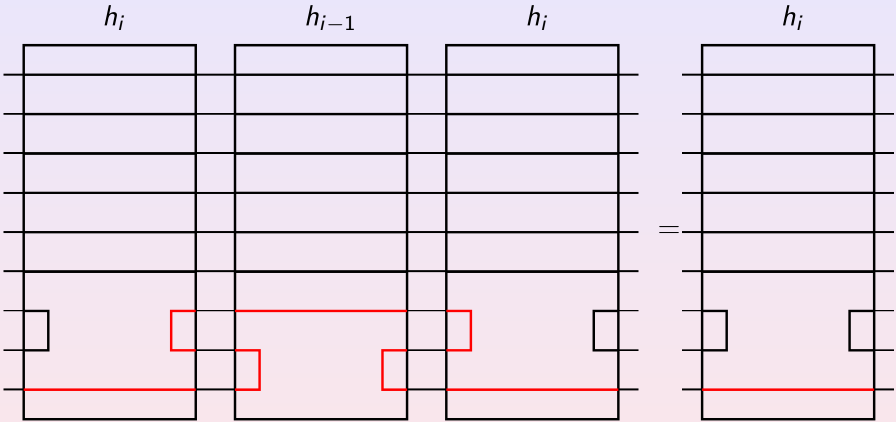
$h_i h_j = h_j h_i$ if $|i - j| \geq 2$



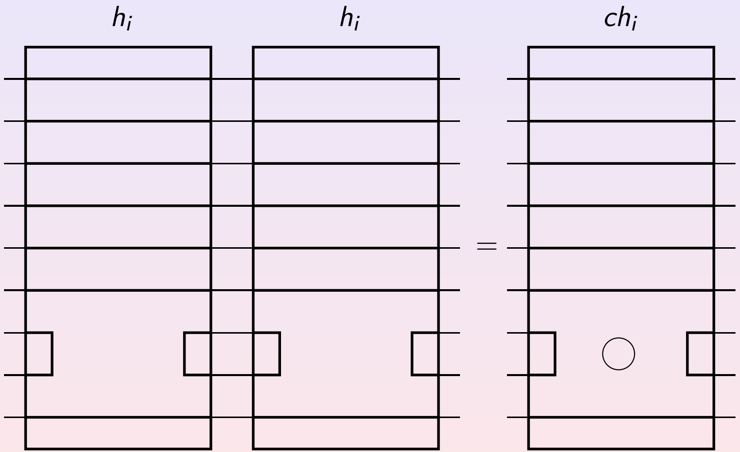
$h_i h_j h_i = h_i$ if $|i - j| = 1$



$h_i h_j h_i = h_i$ if $|i - j| = 1$



$h_i h_i = ch_i$



Kauffman Monoids as Wire Monoids, continued

Thus, the "planar" wire monoid generated by the hooks and the circle satisfies the relations of $\mathcal{K}_n$ and is therefore a homomorphic image of $\mathcal{K}_n$. In fact, this wire monoid is isomorphic to $\mathcal{K}_n$ (requires some work). This connection was realized by Jones who didn't bother himself with a formal proof. Such a proof has first been published by Mirjana Borisavljević, Kosta Došen and Zoran Petrić: Kauffman monoids, J. Knot Theory Ramifications 11: 127–143 (2002).

Similarly, one can show that the Jones monoid of $2n$-pin chips with non-crossing wires without circles is generated by the hooks $h_1, \dots, h_{n-1}$ subject the relations

$$h_ih_j = h_jh_i \quad \text{if } |i-j| \geq 2,$$
$$h_ih_jh_i = h_i \quad \text{if } |i-j| = 1,$$
$$h_ih_i = h_i.$$

Thus, it spans the Temperley–Lieb algebra $\mathcal{TL}_n(1)$.

Kauffman Monoids as Wire Monoids, continued

Thus, the "planar" wire monoid generated by the hooks and the circle satisfies the relations of $\mathcal{K}_n$ and is therefore a homomorphic image of $\mathcal{K}_n$. In fact, this wire monoid is isomorphic to $\mathcal{K}_n$ (requires some work). This connection was realized by Jones who didn't bother himself with a formal proof. Such a proof has first been published by Mirjana Borisavljević, Kosta Došen and Zoran Petrić: Kauffman monoids, J. Knot Theory Ramifications 11: 127–143 (2002).

Similarly, one can show that the Jones monoid of $2n$-pin chips with non-crossing wires without circles is generated by the hooks $h_1, \dots, h_{n-1}$ subject the relations

$$h_i h_j = h_j h_i \quad \text{if } |i-j| \geq 2,$$
$$h_i h_j h_i = h_i \quad \text{if } |i-j| = 1,$$
$$h_i h_i = h_i.$$

Thus, it spans the Temperley–Lieb algebra $\mathcal{TL}_n(1)$.

Kauffman Monoids as Wire Monoids, continued

Thus, the "planar" wire monoid generated by the hooks and the circle satisfies the relations of $\mathcal{K}_n$ and is therefore a homomorphic image of $\mathcal{K}_n$. In fact, this wire monoid is isomorphic to $\mathcal{K}_n$ (requires some work). This connection was realized by Jones who didn't bother himself with a formal proof. Such a proof has first been published by Mirjana Borisavljević, Kosta Došen and Zoran Petrić: Kauffman monoids, J. Knot Theory Ramifications 11: 127–143 (2002).

Similarly, one can show that the Jones monoid of $2n$-pin chips with non-crossing wires without circles is generated by the hooks $h_1, \ldots, h_{n-1}$ subject the relations

$$\begin{aligned} h_i h_j &= h_j h_i \quad \text{if } |i-j| \geq 2, \\ h_i h_j h_i &= h_i \quad \text{if } |i-j| = 1, \\ h_i h_i &= h_i. \end{aligned}$$

Thus, it spans the Temperley–Lieb algebra $\mathcal{TL}_n(1)$.

Kauffman Monoids as Wire Monoids, continued

Thus, the "planar" wire monoid generated by the hooks and the circle satisfies the relations of $\mathcal{K}_n$ and is therefore a homomorphic image of $\mathcal{K}_n$. In fact, this wire monoid is isomorphic to $\mathcal{K}_n$ (requires some work). This connection was realized by Jones who didn't bother himself with a formal proof. Such a proof has first been published by Mirjana Borisavljević, Kosta Došen and Zoran Petrić: Kauffman monoids, J. Knot Theory Ramifications 11: 127–143 (2002).

Similarly, one can show that the Jones monoid of $2n$-pin chips with non-crossing wires without circles is generated by the hooks $h_1, \dots, h_{n-1}$ subject the relations

$$\begin{aligned} h_i h_j &= h_j h_i \quad \text{if } |i-j| \geq 2, \\ h_i h_j h_i &= h_i \quad \text{if } |i-j| = 1, \\ h_i h_i &= h_i. \end{aligned}$$

Thus, it spans the Temperley–Lieb algebra $\mathcal{TL}_n(1)$.

Kauffman Monoids as Wire Monoids, continued

Thus, the "planar" wire monoid generated by the hooks and the circle satisfies the relations of $\mathcal{K}_n$ and is therefore a homomorphic image of $\mathcal{K}_n$. In fact, this wire monoid is isomorphic to $\mathcal{K}_n$ (requires some work). This connection was realized by Jones who didn't bother himself with a formal proof. Such a proof has first been published by Mirjana Borisavljević, Kosta Došen and Zoran Petrić: Kauffman monoids, J. Knot Theory Ramifications 11: 127–143 (2002).

Similarly, one can show that the Jones monoid of $2n$-pin chips with non-crossing wires without circles is generated by the hooks $h_1, \ldots, h_{n-1}$ subject the relations

$$
\begin{aligned}
h_i h_j &= h_j h_i \quad \text{if } |i-j| \geq 2,\\
h_i h_j h_i &= h_i \quad \text{if } |i-j| = 1,\\
h_i h_i &= h_i.
\end{aligned}
$$

Thus, it spans the Temperley–Lieb algebra $\mathcal{TL}_n(1)$.

Kauffman Monoids as Wire Monoids, continued

Thus, the "planar" wire monoid generated by the hooks and the circle satisfies the relations of $\mathcal{K}_n$ and is therefore a homomorphic image of $\mathcal{K}_n$. In fact, this wire monoid is isomorphic to $\mathcal{K}_n$ (requires some work). This connection was realized by Jones who didn't bother himself with a formal proof. Such a proof has first been published by Mirjana Borisavljević, Kosta Došen and Zoran Petrić: Kauffman monoids, J. Knot Theory Ramifications 11: 127–143 (2002).

Similarly, one can show that the Jones monoid of $2n$-pin chips with non-crossing wires without circles is generated by the hooks $h_1, \dots, h_{n-1}$ subject the relations

$$\begin{aligned} h_i h_j &= h_j h_i \quad \text{if } |i-j| \geq 2, \\ h_i h_j h_i &= h_i \quad \text{if } |i-j| = 1, \\ h_i h_i &= h_i. \end{aligned}$$

Thus, it spans the Temperley–Lieb algebra $\mathcal{TL}_n(1)$.

Identities in Kauffman Monoids

The Kauffman monoid $\mathcal{K}_n$ is infinite (due to circles).

The Kauffman monoid $\mathcal{K}_2$ is commutative, and thus, finitely based. Hence we have a complete solution to the finite basis problem for the Kauffman monoids.

But how can one recognize the identities that hold in $\mathcal{K}_n$, $n \geq 3$?

The theorem above was obtained via a 'high-level' argument that allows one to prove, under certain conditions, that a semigroup $\mathcal{S}$ admits no finite identity basis, without writing down any concrete identity holding in $\mathcal{S}$! Thus, no information about the identities of $\mathcal{K}_n$ for $n \geq 3$ can be extracted from the proof, besides the mere fact that non-trivial identities in $\mathcal{K}_n$ do exist.

Identities in Kauffman Monoids

The Kauffman monoid $\mathcal{K}_n$ is infinite (due to circles).

Theorem (Karl Auinger, Yuzhu Chen, Xun Hu, Yanfeng Luo, and V., The finite basis problem for Kauffman monoids, Algebra Universalis 74(3-4): 333–350 (2015))

For each $n \geq 3$, the Kauffman monoid $\mathcal{K}_n$ is nonfinitely based.

The Kauffman monoid $\mathcal{K}_2$ is commutative, and thus, finitely based. Hence we have a complete solution to the finite basis problem for the Kauffman monoids.

But how can one recognize the identities that hold in $\mathcal{K}_n$, $n \geq 3$?

The theorem above was obtained via a 'high-level' argument that allows one to prove, under certain conditions, that a semigroup $\mathcal{S}$ admits no finite identity basis, without writing down any concrete identity holding in $\mathcal{S}$! Thus, no information about the identities of $\mathcal{K}_n$ for $n \geq 3$ can be extracted from the proof, besides the mere fact that non-trivial identities in $\mathcal{K}_n$ do exist.

Identities in Kauffman Monoids

The Kauffman monoid $\mathcal{K}_n$ is infinite (due to circles).

Theorem (Karl Auinger, Yuzhu Chen, Xun Hu, Yanfeng Luo, and V., The finite basis problem for Kauffman monoids, Algebra Universalis 74(3-4): 333–350 (2015))

For each $n \geq 3$, the Kauffman monoid $\mathcal{K}_n$ is nonfinitely based.

The Kauffman monoid $\mathcal{K}_2$ is commutative, and thus, finitely based. Hence we have a complete solution to the finite basis problem for the Kauffman monoids.

But how can one recognize the identities that hold in $\mathcal{K}_n$, $n \geq 3$?

The theorem above was obtained via a 'high-level' argument that allows one to prove, under certain conditions, that a semigroup $\mathcal{S}$ admits no finite identity basis, without writing down any concrete identity holding in $\mathcal{S}$! Thus, no information about the identities of $\mathcal{K}_n$ for $n \geq 3$ can be extracted from the proof, besides the mere fact that non-trivial identities in $\mathcal{K}_n$ do exist.

Identities in Kauffman Monoids

The Kauffman monoid $\mathcal{K}_n$ is infinite (due to circles).

Theorem (Karl Auinger, Yuzhu Chen, Xun Hu, Yanfeng Luo, and V., The finite basis problem for Kauffman monoids, Algebra Universalis 74(3-4): 333–350 (2015))

For each $n \geq 3$, the Kauffman monoid $\mathcal{K}_n$ is nonfinitely based.

The Kauffman monoid $\mathcal{K}_2$ is commutative, and thus, finitely based. Hence we have a complete solution to the finite basis problem for the Kauffman monoids.

But how can one recognize the identities that hold in $\mathcal{K}_n$, $n \geq 3$?

The theorem above was obtained via a 'high-level' argument that allows one to prove, under certain conditions, that a semigroup $\mathcal{S}$ admits no finite identity basis, without writing down any concrete identity holding in $\mathcal{S}$! Thus, no information about the identities of $\mathcal{K}_n$ for $n \geq 3$ can be extracted from the proof, besides the mere fact that non-trivial identities in $\mathcal{K}_n$ do exist.

Identities in Kauffman Monoids

The Kauffman monoid $\mathcal{K}_n$ is infinite (due to circles).

Theorem (Karl Auinger, Yuzhu Chen, Xun Hu, Yanfeng Luo, and V., The finite basis problem for Kauffman monoids, Algebra Universalis 74(3-4): 333–350 (2015))

For each $n \geq 3$, the Kauffman monoid $\mathcal{K}_n$ is nonfinitely based.

The Kauffman monoid $\mathcal{K}_2$ is commutative, and thus, finitely based. Hence we have a complete solution to the finite basis problem for the Kauffman monoids.

But how can one recognize the identities that hold in $\mathcal{K}_n$, $n \geq 3$?

The theorem above was obtained via a 'high-level' argument that allows one to prove, under certain conditions, that a semigroup $\mathcal{S}$ admits no finite identity basis, without writing down any concrete identity holding in $\mathcal{S}$! Thus, no information about the identities of $\mathcal{K}_n$ for $n \geq 3$ can be extracted from the proof, besides the mere fact that non-trivial identities in $\mathcal{K}_n$ do exist.

Identities in Kauffman Monoids

The Kauffman monoid $\mathcal{K}_n$ is infinite (due to circles).

Theorem (Karl Auinger, Yuzhu Chen, Xun Hu, Yanfeng Luo, and V., The finite basis problem for Kauffman monoids, Algebra Universalis 74(3-4): 333–350 (2015))

For each $n \geq 3$, the Kauffman monoid $\mathcal{K}_n$ is nonfinitely based.

The Kauffman monoid $\mathcal{K}_2$ is commutative, and thus, finitely based. Hence we have a complete solution to the finite basis problem for the Kauffman monoids.

But how can one recognize the identities that hold in $\mathcal{K}_n$, $n \geq 3$?

The theorem above was obtained via a 'high-level' argument that allows one to prove, under certain conditions, that a semigroup $\mathcal{S}$ admits no finite identity basis, without writing down any concrete identity holding in $\mathcal{S}$! Thus, no information about the identities of $\mathcal{K}_n$ for $n \geq 3$ can be extracted from the proof, besides the mere fact that non-trivial identities in $\mathcal{K}_n$ do exist.

Identities in Kauffman Monoids

The Kauffman monoid $\mathcal{K}_n$ is infinite (due to circles).

Theorem (Karl Auinger, Yuzhu Chen, Xun Hu, Yanfeng Luo, and V., The finite basis problem for Kauffman monoids, Algebra Universalis 74(3-4): 333–350 (2015))

For each $n \geq 3$, the Kauffman monoid $\mathcal{K}_n$ is nonfinitely based.

The Kauffman monoid $\mathcal{K}_2$ is commutative, and thus, finitely based. Hence we have a complete solution to the finite basis problem for the Kauffman monoids.

But how can one recognize the identities that hold in $\mathcal{K}_n$, $n \geq 3$?

The theorem above was obtained via a 'high-level' argument that allows one to prove, under certain conditions, that a semigroup $\mathcal{S}$ admits no finite identity basis, without writing down any concrete identity holding in $\mathcal{S}$! Thus, no information about the identities of $\mathcal{K}_n$ for $n \geq 3$ can be extracted from the proof, besides the mere fact that non-trivial identities in $\mathcal{K}_n$ do exist.

Identities in $\mathcal{K}_3$

A jump is a triple (x, C, y), where x and y are (not necessarily distinct) letters and C is a (possibly empty) set of letters that contains neither x nor y. The jump (x, C, y) occurs in a word w if w can be decomposed as $w = uxtyv$ where u, t, v are (possibly empty) words and $C = \mathrm{alph}(t)$. The first (last) occurrence sequence of a word w is obtained from w by retaining only the first (respectively, the last) occurrence of each letter from $\mathrm{alph}(w)$.

Identities in $\mathcal{K}_3$

A jump is a triple (x, C, y), where x and y are (not necessarily distinct) letters and C is a (possibly empty) set of letters that contains neither x nor y. The jump (x, C, y) occurs in a word w if w can be decomposed as $w = uxtyv$ where u, t, v are (possibly empty) words and $C = \mathrm{alph}(t)$.

The first (last) occurrence sequence of a word w is obtained from w by retaining only the first (respectively, the last) occurrence of each letter from $\mathrm{alph}(w)$.

A jump is a triple (x, C, y), where x and y are (not necessarily distinct) letters and C is a (possibly empty) set of letters that contains neither x nor y. The jump (x, C, y) occurs in a word w if w can be decomposed as $w = uxtyv$ where u, t, v are (possibly empty) words and $C = \mathrm{alph}(t)$. For instance, the jump $(x, \{y, z\}, x)$ occurs twice in the word xy^2zxzy^2x.

The first (last) occurrence sequence of a word w is obtained from w by retaining only the first (respectively, the last) occurrence of each letter from $\mathrm{alph}(w)$.

Identities in $\mathcal{K}_3$

A jump is a triple (x, C, y), where x and y are (not necessarily distinct) letters and C is a (possibly empty) set of letters that contains neither x nor y. The jump (x, C, y) occurs in a word w if w can be decomposed as $w = uxtyv$ where u, t, v are (possibly empty) words and $C = \mathrm{alph}(t)$. For instance, the jump $(x, \{y, z\}, x)$ occurs twice in the word xy^2zxzy^2x. Here is the first occurrence: xy^2zxzy^2x.

The first (last) occurrence sequence of a word w is obtained from w by retaining only the first (respectively, the last) occurrence of each letter from $\mathrm{alph}(w)$.

Identities in $\mathcal{K}_3$

A jump is a triple (x, C, y), where x and y are (not necessarily distinct) letters and C is a (possibly empty) set of letters that contains neither x nor y. The jump (x, C, y) occurs in a word w if w can be decomposed as $w = uxtyv$ where u, t, v are (possibly empty) words and $C = \text{alph}(t)$. For instance, the jump $(x, \{y, z\}, x)$ occurs twice in the word xy^2zxzy^2x. Here is the first occurrence: xy^2zxzy^2x. And here is the second one: xy^2zxzy^2x.

The first (last) occurrence sequence of a word w is obtained from w by retaining only the first (respectively, the last) occurrence of each letter from $\text{alph}(w)$.

Identities in $\mathcal{K}_3$

A jump is a triple (x, C, y), where x and y are (not necessarily distinct) letters and C is a (possibly empty) set of letters that contains neither x nor y. The jump (x, C, y) occurs in a word w if w can be decomposed as $w = uxtyv$ where u, t, v are (possibly empty) words and $C = \mathrm{alph}(t)$.
The first (last) occurrence sequence of a word w is obtained from w by retaining only the first (respectively, the last) occurrence of each letter from $\mathrm{alph}(w)$.

Identities in $\mathcal{K}_3$

A jump is a triple (x, C, y), where x and y are (not necessarily distinct) letters and C is a (possibly empty) set of letters that contains neither x nor y. The jump (x, C, y) occurs in a word w if w can be decomposed as $w = uxtyv$ where u, t, v are (possibly empty) words and $C = \mathrm{alph}(t)$.
The first (last) occurrence sequence of a word w is obtained from w by retaining only the first (respectively, the last) occurrence of each letter from $\mathrm{alph}(w)$.

Theorem (Yuzhu Chen, Xun Hu, Nikita Kitov, Yanfeng Luo, and V., Identities of the Kauffman Monoid $\mathcal{K}_3$, Comm. Algebra 48(5): 1956–1968 (2020))

An identity $w \bumpeq w'$ holds in the Kauffman monoid $\mathcal{K}_3$ iff
1) w and w' have the same first occurrence sequence and the same last occurrence sequence, and
2) every jump occurs the same number of times in w and w'.

Example and Corollary

For instance, the identity $x^2yx \doteq xyx^2$ holds in $\mathcal{K}_3$.
The first occurrence sequence of x^2yx and xyx^2 is xy,
the last occurrence sequence of x^2yx and xyx^2 is yx,
and the jumps that occur in x^2yx and xyx^2 are
$(x, \varnothing, x)$, $(x, \varnothing, y)$, $(x, \{y\}, x)$, and $(y, \varnothing, x)$,
each occurring exactly once.

Example and Corollary

For instance, the identity $x^2yx \doteq xyx^2$ holds in $\mathcal{K}_3$.
The first occurrence sequence of x^2yx and xyx^2 is xy,
the last occurrence sequence of x^2yx and xyx^2 is yx,
and the jumps that occur in x^2yx and xyx^2 are
$(x, \varnothing, x)$, $(x, \varnothing, y)$, $(x, \{y\}, x)$, and $(y, \varnothing, x)$,
each occurring exactly once.

Example and Corollary

For instance, the identity $x^2yx \doteq xyx^2$ holds in $\mathcal{K}_3$.
The first occurrence sequence of x^2yx and xyx^2 is xy,
the last occurrence sequence of x^2yx and xyx^2 is yx,
and the jumps that occur in x^2yx and xyx^2 are
$(x, \varnothing, x)$, $(x, \varnothing, y)$, $(x, \{y\}, x)$, and $(y, \varnothing, x)$,
each occurring exactly once.

Example and Corollary

For instance, the identity $x^2yx \simeq xyx^2$ holds in $\mathcal{K}_3$.
The first occurrence sequence of x^2yx and xyx^2 is xy,
the last occurrence sequence of x^2yx and xyx^2 is yx,
and the jumps that occur in x^2yx and xyx^2 are
$(x, \varnothing, x)$, $(x, \varnothing, y)$, $(x, \{y\}, x)$, and $(y, \varnothing, x)$,
each occurring exactly once.

Example and Corollary

For instance, the identity $x^2yx \doteq xyx^2$ holds in $\mathcal{K}_3$.
The first occurrence sequence of x^2yx and xyx^2 is xy,
the last occurrence sequence of x^2yx and xyx^2 is yx,
and the jumps that occur in x^2yx and xyx^2 are
$(x, \varnothing, x)$, $(x, \varnothing, y)$, $(x, \{y\}, x)$, and $(y, \varnothing, x)$,
each occurring exactly once.

One can check the conditions of the above theorem in $O(|ww'|\log|ww'|)$ time. Hence:

Corollary

The problem $\textsc{Check-Id}(\mathcal{K}_3)$ lies in the complexity class P.

Identities in $\mathcal{K}_4$

The next step is, of course, to consider the identities of $\mathcal{K}_4$.

For some time, we tried to show that the identity $x^2yx \doteq xyx^2$ fails in $\mathcal{K}_4$.
We did not succeed, which was a sort of surprise because, informally speaking, $\mathcal{K}_4$ is much more complicated than $\mathcal{K}_3$.

Identities in $\mathcal{K}_4$

The next step is, of course, to consider the identities of $\mathcal{K}_4$.
For some time, we tried to show that the identity $x^2yx \doteq xyx^2$ fails in $\mathcal{K}_4$.
We did not succeed, which was a sort of surprise because, informally speaking, $\mathcal{K}_4$ is much more complicated than $\mathcal{K}_3$.

Identities in $\mathcal{K}_4$

The next step is, of course, to consider the identities of $\mathcal{K}_4$.
For some time, we tried to show that the identity $x^2yx \doteq xyx^2$ fails in $\mathcal{K}_4$.
We did not succeed, which was a sort of surprise because, informally speaking, $\mathcal{K}_4$ is much more complicated than $\mathcal{K}_3$.

Identities in $\mathcal{K}_4$

The next step is, of course, to consider the identities of $\mathcal{K}_4$.
For some time, we tried to show that the identity $x^2yx \doteq xyx^2$ fails in $\mathcal{K}_4$.
We did not succeed, which was a sort of surprise because, informally speaking, $\mathcal{K}_4$ is much more complicated than $\mathcal{K}_3$.

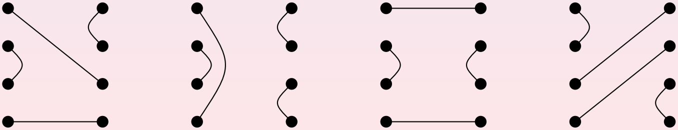
Here are the four 'basic' chips from $\mathcal{K}_3$ (all others chips in $\mathcal{K}_3$ are obtained by adding circles to these four and the unit chip $\equiv$):

Identities in $\mathcal{K}_4$

The next step is, of course, to consider the identities of $\mathcal{K}_4$.
For some time, we tried to show that the identity $x^2yx \bumpeq xyx^2$ fails in $\mathcal{K}_4$.
We did not succeed, which was a sort of surprise because, informally speaking, $\mathcal{K}_4$ is much more complicated than $\mathcal{K}_3$.

Here are the four 'basic' chips from $\mathcal{K}_3$ (all others chips in $\mathcal{K}_3$ are obtained by adding circles to these four and the unit chip $\equiv$):

To compare, here are a few (not all!) basic chips from $\mathcal{K}_4$:



Identities in $\mathcal{K}_4$, continued

Finally, we have obtained the following result, which I still find somewhat counterintuitive:

Theorem (Nikita Kitov and V., see the Yurifest volume)

The monoids $\mathcal{K}_3$ and $\mathcal{K}_4$ satisfy the same identities.

The nature of the proof is geometric: we use certain properties of the following 'surgery' map on the basic chips of $\mathcal{K}_4$:

Identities in $\mathcal{K}_4$, continued

Finally, we have obtained the following result, which I still find somewhat counterintuitive:

Theorem (Nikita Kitov and V., see the Yurifest volume)

The monoids $\mathcal{K}_3$ and $\mathcal{K}_4$ satisfy the same identities.

Corollary

The problem $\textsc{Check-Id}(\mathcal{K}_4)$ lies in the complexity class P.

The nature of the proof is geometric: we use certain properties of the following 'surgery' map on the basic chips of $\mathcal{K}_4$:

Identities in $\mathcal{K}_4$, continued

Finally, we have obtained the following result, which I still find somewhat counterintuitive:

Theorem (Nikita Kitov and V., see the Yurifest volume)

The monoids $\mathcal{K}_3$ and $\mathcal{K}_4$ satisfy the same identities.

The nature of the proof is geometric: we use certain properties of the following 'surgery' map on the basic chips of $\mathcal{K}_4$:

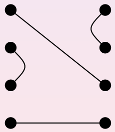
Identities in $\mathcal{K}_4$, continued

Finally, we have obtained the following result, which I still find somewhat counterintuitive:

Theorem (Nikita Kitov and V., see the Yurifest volume)

The monoids $\mathcal{K}_3$ and $\mathcal{K}_4$ satisfy the same identities.

The nature of the proof is geometric: we use certain properties of the following 'surgery' map on the basic chips of $\mathcal{K}_4$:



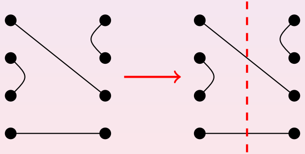
Identities in $\mathcal{K}_4$, continued

Finally, we have obtained the following result, which I still find somewhat counterintuitive:

Theorem (Nikita Kitov and V., see the Yurifest volume)

The monoids $\mathcal{K}_3$ and $\mathcal{K}_4$ satisfy the same identities.

The nature of the proof is geometric: we use certain properties of the following 'surgery' map on the basic chips of $\mathcal{K}_4$:



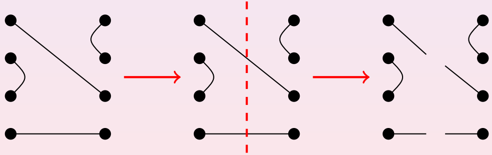
Identities in $\mathcal{K}_4$, continued

Finally, we have obtained the following result, which I still find somewhat counterintuitive:

Theorem (Nikita Kitov and V., see the Yurifest volume)

The monoids $\mathcal{K}_3$ and $\mathcal{K}_4$ satisfy the same identities.

The nature of the proof is geometric: we use certain properties of the following 'surgery' map on the basic chips of $\mathcal{K}_4$:



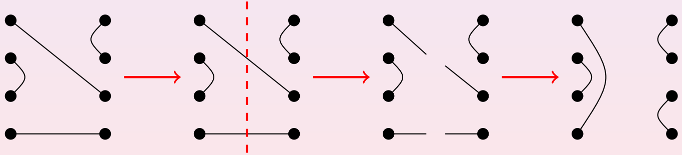
Identities in $\mathcal{K}_4$, continued

Finally, we have obtained the following result, which I still find somewhat counterintuitive:

Theorem (Nikita Kitov and V., see the Yurifest volume)

The monoids $\mathcal{K}_3$ and $\mathcal{K}_4$ satisfy the same identities.

The nature of the proof is geometric: we use certain properties of the following 'surgery' map on the basic chips of $\mathcal{K}_4$:



Identities in $\mathcal{K}_5$?

Well, the next step is to consider the identities of $\mathcal{K}_5$.

Does the identity $x^2yx \doteq xyx^2$ hold in $\mathcal{K}_5$? No, it fails, e.g., under $x \mapsto h_1h_2h_3$, $y \mapsto h_4$.

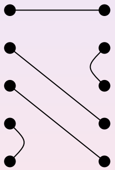
Identities in $\mathcal{K}_5$?

Well, the next step is to consider the identities of $\mathcal{K}_5$.
Does the identity $x^2yx \doteq xyx^2$ hold in $\mathcal{K}_5$? No, it fails, e.g., under $x \mapsto h_1h_2h_3$, $y \mapsto h_4$.

Identities in $\mathcal{K}_5$?

Well, the next step is to consider the identities of $\mathcal{K}_5$.
Does the identity $x^2yx \doteq xyx^2$ hold in $\mathcal{K}_5$? No, it fails,
e.g., under $x \mapsto h_1h_2h_3$, $y \mapsto h_4$.

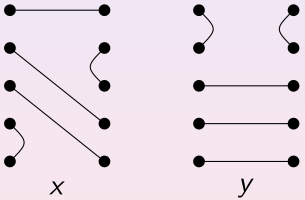
Well, the next step is to consider the identities of $\mathcal{K}_5$.
Does the identity $x^2yx \doteq xyx^2$ hold in $\mathcal{K}_5$? No, it fails,
e.g., under $x \mapsto h_1h_2h_3$, $y \mapsto h_4$.



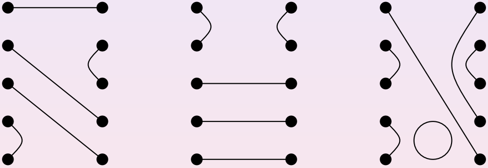
x

Identities in $\mathcal{K}_5$?

Well, the next step is to consider the identities of $\mathcal{K}_5$.
Does the identity $x^2yx \doteq xyx^2$ hold in $\mathcal{K}_5$? No, it fails, e.g., under $x \mapsto h_1h_2h_3$, $y \mapsto h_4$.

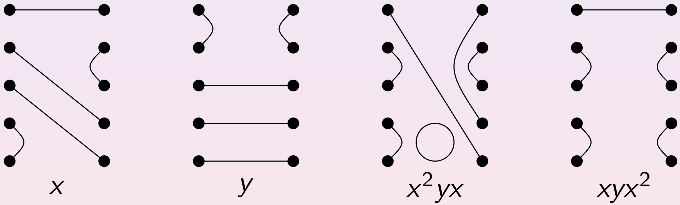


Well, the next step is to consider the identities of $\mathcal{K}_5$.
Does the identity $x^2yx \simeq xyx^2$ hold in $\mathcal{K}_5$? No, it fails, e.g., under $x \mapsto h_1h_2h_3$, $y \mapsto h_4$.



x $\quad$ y $\quad$ x^2yx

Well, the next step is to consider the identities of $\mathcal{K}_5$.
Does the identity $x^2yx \doteq xyx^2$ hold in $\mathcal{K}_5$? No, it fails, e.g., under $x \mapsto h_1h_2h_3$, $y \mapsto h_4$.



Open Problems

- Checking identities in $\mathcal{K}_n$, $n > 4$
- Checking identities in other interesting infinite monoids, e.g., monoids of tropical matrices, cobordism monoids, plactic monoids, etc.
- Finding natural semigroups $\mathcal{S}$ such that $\textsc{Check-Id}(\mathcal{S})$ would be complete for various complexity classes, e.g., NP, PSPACE, EXPTIME, etc.
- Similar problems for checking quasi-identities

Open Problems

- Checking identities in $\mathcal{K}_n$, $n > 4$
- Checking identities in other interesting infinite monoids, e.g., monoids of tropical matrices, cobordism monoids, plactic monoids, etc.
- Finding natural semigroups $\mathcal{S}$ such that $\textsc{Check-Id}(\mathcal{S})$ would be complete for various complexity classes, e.g., NP, PSPACE, EXPTIME, etc.
- Similar problems for checking quasi-identities

Open Problems

- Checking identities in $\mathcal{K}_n$, $n > 4$
- Checking identities in other interesting infinite monoids, e.g., monoids of tropical matrices, cobordism monoids, plactic monoids, etc.
- Finding natural semigroups $\mathcal{S}$ such that $\textsc{Check-Id}(\mathcal{S})$ would be complete for various complexity classes, e.g., NP, PSPACE, EXPTIME, etc.
- Similar problems for checking quasi-identities

Open Problems

- Checking identities in $\mathcal{K}_n$, $n > 4$
- Checking identities in other interesting infinite monoids, e.g., monoids of tropical matrices, cobordism monoids, plactic monoids, etc.
- Finding natural semigroups $\mathcal{S}$ such that $\textsc{Check-Id}(\mathcal{S})$ would be complete for various complexity classes, e.g., NP, PSPACE, EXPTIME, etc.
- Similar problems for checking quasi-identities