Being clever with semigroup calculations

Wilf Wilson
University of St Andrews

18th August 2015

Me

- First-year maths PhD student at St Andrews.

What do I do?

- Good question: hopefully this talk will give an idea.

What do I do?

- Good question: hopefully this talk will give an idea.
- I come up with ways to efficiently calculate things about semigroups.
  - ‘Computational semigroup theory’.

Definitions (not too important)

Definition (Semigroup)

A semigroup is a set with an associative binary operation.

- Finite for today.

Definitions (not too important)

Definition (Semigroup)

A semigroup is a set with an associative binary operation.

- Finite for today.
- Can think of semigroup as a set of finite transformations

$$f : \{1, 2, \ldots, n\} \to \{1, 2, \ldots, n\}.$$

which is closed under composition, e.g. $\left\{ \begin{pmatrix} 1 & 2 & 3 \\ 1 & 2 & 3 \end{pmatrix}, \begin{pmatrix} 1 & 2 & 3 \\ 1 & 1 & 1 \end{pmatrix} \right\}$.

Definitions (not too important)

Definition (Semigroup)

A semigroup is a set with an associative binary operation.

- Finite for today.
- Can think of semigroup as a set of finite transformations

$$f : \{1, 2, \ldots, n\} \to \{1, 2, \ldots, n\}.$$

  which is closed under composition, e.g. $\left\{ \begin{pmatrix} 1 & 2 & 3 \\ 1 & 2 & 3 \end{pmatrix}, \begin{pmatrix} 1 & 2 & 3 \\ 1 & 1 & 1 \end{pmatrix} \right\}$.

- **Key point**: a semigroup will be defined by its generators.
  - i.e. $S := \langle f, g \rangle$ for some transformations f and g.
  - There are n^n transformations of the set $\{1, 2, \ldots, n\}$.

GAP demonstration: group theory

GAP demonstration: group theory

```
gap> a := (1,2)(3,10)(4,16,6,12,8,14)(5,11,7,13,9,15);;
gap> b := (1,10,7)(2,3,15)(4,13,9)(5,11,12);;
gap> G := Group(a, b);;
```

GAP demonstration: group theory

```
gap> a := (1,2)(3,10)(4,16,6,12,8,14)(5,11,7,13,9,15);;
gap> b := (1,10,7)(2,3,15)(4,13,9)(5,11,12);;
gap> G := Group(a, b);;
gap> StructureDescription(G);
"SL(2,7)"
```

GAP demonstration: group theory

```
gap> a := (1,2)(3,10)(4,16,6,12,8,14)(5,11,7,13,9,15);;
gap> b := (1,10,7)(2,3,15)(4,13,9)(5,11,12);;
gap> G := Group(a, b);;
gap> StructureDescription(G);
"SL(2,7)"
gap> Size(G);
336
```

GAP demonstration: group theory

```
gap> a := (1,2)(3,10)(4,16,6,12,8,14)(5,11,7,13,9,15);;
gap> b := (1,10,7)(2,3,15)(4,13,9)(5,11,12);;
gap> G := Group(a, b);;
gap> StructureDescription(G);
"SL(2,7)"
gap> Size(G);
336
gap> IsSimpleGroup(G);
false
```

GAP demonstration: group theory

```
gap> a := (1,2)(3,10)(4,16,6,12,8,14)(5,11,7,13,9,15);;
gap> b := (1,10,7)(2,3,15)(4,13,9)(5,11,12);;
gap> G := Group(a, b);;
gap> StructureDescription(G);
"SL(2,7)"
gap> Size(G);
336
gap> IsSimpleGroup(G);
false
gap> H := Stabilizer(G, 4);;
gap> StructureDescription(H);
"C7 : C3"
```

GAP demonstration: group theory

```
gap> a := (1,2)(3,10)(4,16,6,12,8,14)(5,11,7,13,9,15);;
gap> b := (1,10,7)(2,3,15)(4,13,9)(5,11,12);;
gap> G := Group(a, b);;
gap> StructureDescription(G);
"SL(2,7)"
gap> Size(G);
336
gap> IsSimpleGroup(G);
false
gap> H := Stabilizer(G, 4);;
gap> StructureDescription(H);
"C7 : C3"
gap> IsNormal(G, H);
false
```

GAP demonstration: group theory

```
gap> a := (1,2)(3,10)(4,16,6,12,8,14)(5,11,7,13,9,15);;
gap> b := (1,10,7)(2,3,15)(4,13,9)(5,11,12);;
gap> G := Group(a, b);;
gap> StructureDescription(G);
"SL(2,7)"
gap> Size(G);
336
gap> IsSimpleGroup(G);
false
gap> H := Stabilizer(G, 4);;
gap> StructureDescription(H);
"C7 : C3"
gap> IsNormal(G, H);
false
gap> F := H / DerivedSubgroup(H);;
gap> StructureDescription(F);
"C3"
```

Computational group theory

- It's well developed, mature: 50+ years.
- Fast, polynomial-time algorithms.
- Randomised algorithms.

Computational semigroup theory

- ~~It's well developed, mature: 50+ years~~. New.
- Fast, polynomial-time algorithms.
- Randomised algorithms.

Computational semigroup theory

- ~~It's well developed, mature: 50+ years~~. New.
- ~~Fast, polynomial-time algorithms~~. Often expontential complexity.
- Randomised algorithms.

Computational semigroup theory

- ~~It's well developed, mature: 50+ years~~. New.
- ~~Fast, polynomial-time algorithms~~. Often expontential complexity.
- ~~Randomised algorithms~~. No randomised algorithms.

Computational semigroup theory

- ~~It's well developed, mature: 50+ years~~. New.
- ~~Fast, polynomial-time algorithms~~. Often expontential complexity.
- ~~Randomised algorithms~~. No randomised algorithms.

But we still want to do it!

Semigroups are just harder

The number of groups of order 10:

Semigroups are just harder

The number of groups of order 10: 2.

Semigroups are just harder

The number of groups of order 10: 2.

The number of semigroups of order 10:

Semigroups are just harder

The number of groups of order 10: 2.

The number of semigroups of order 10: 12,418,001,077,381,302,684.

Semigroups are just harder

The number of groups of order 10: 2.

The number of semigroups of order 10: 12,418,001,077,381,302,684.

We can never hope to do as well as groups.

Use the generators: Commutativity

Semigroup: $S = \langle f_1, f_2, \ldots, f_k \rangle$.

How do we check if S is commutative?

Use the generators: Commutativity

Semigroup: $S = \langle f_1, f_2, \ldots, f_k \rangle$.

How do we check if S is commutative? Easy!

- Every element is a product of the f_i.
- If the generators commute, the semigroup commutes (and vice versa).

- So we just need to check the generators.

Use theorems: Nilpotency

Definition (Nilpotent)

A semigroup is nilpotent if each element has some power equal to 0.

Use theorems: Nilpotency

Definition (Nilpotent)

A semigroup is nilpotent if each element has some power equal to 0.

Definition (Nilpotency degree)

For a nilpotent semigroup, the nilpotency degree is the least positive r such that every product of r elements equals 0.

Use theorems: Nilpotency

Definition (Nilpotent)

A semigroup is nilpotent if each element has some power equal to 0.

Definition (Nilpotency degree)

For a nilpotent semigroup, the nilpotency degree is the least positive r such that every product of r elements equals 0.

Theorems:

- Nilpotent $\Leftrightarrow$ there is a 0 but no other idempotents.
- Nilpotency degree $=$ length of longest chain of principal ideals $+\ 1$.

Use what you already know: Synchronizing

Definition (Synchronizing semigroup)

A semigroup is synchronizing if it contains a constant function.

- A constant map e is right-zero, i.e. it satisfies $xe = e$ for all x.
- If you know your semigroup has a zero, just look at that.
- If you have computed the minimal ideal, then just look in there.

Find equivalent definitions: Inverse semigroups

Definition (An inverse)

An element x has an inverse y if $xyx = x$.

Find equivalent definitions: Inverse semigroups

Definition (An inverse)

An element x has an inverse y if $xyx = x$.

Definition (Inverse semigroup)

A semigroup is inverse if every element has a **unique** inverse.

Find equivalent definitions: Inverse semigroups

Definition (An inverse)

An element x has an <u>inverse</u> y if $xyx = x$.

Definition (Inverse semigroup)

A semigroup is <u>inverse</u> if every element has a **unique** inverse.

For a semigroup S, the following are equivalent:

- S is inverse (every element has exactly one inverse).
- Every element has <u>an</u> inverse and the idempotents commute.
- Every $\mathscr{R}$-class and $\mathscr{L}$-class contains exactly one idempotent.

Code-in implications: "True methods" in GAP

Band	$\Rightarrow$ regular.
Simple and $\mathscr{H}$-trivial	$\Rightarrow$ rectangular band.
Inverse semigroup and $\mathscr{D}$-trivial	$\Rightarrow$ semilattice.
Zero semigroup	$\Rightarrow$ commutative.
Regular and commutative	$\Rightarrow$ inverse.

Difficult things to calculate

Pretty much everything.

- Checking isomorphism between two semigroups.
- Finding the endomorphism monoid of a semigroup.
- Minimal degree transformation representation of a semigroup.

Difficult things to calculate

Pretty much everything.

- Checking isomorphism between two semigroups.
- Finding the endomorphism monoid of a semigroup.
- Minimal degree transformation representation of a semigroup.
    - Useful, and what I want to think about next.

Practical remarks

- GAP: find it at `gap-system.org`
- Semigroups package for GAP.

Practical remarks

End.