Gödel's functional interpretation in constructive algebra

Thomas Powell
Technische Universität Darmstadt

(jww Peter Schuster and Franziskus Wiesnet)

ALGEBRA AND ALGORITHMS
Djerba, Tunisia
4 February 2020

Introduction

This talk is about the extraction of programs from proofs:

PROOFS → PROGRAMS

More specifically, for us:

- PROOFS = nonconstructive maximality arguments from commutative algebra
- PROGRAMS = state-based sequential algorithms

Our main technique for extraction will be:

Gödel's functional ("Dialectica") interpretation + **some cleverness**

Introduction

This talk is about the extraction of programs from proofs:

PROOFS → PROGRAMS

More specifically, for us:

- PROOFS = nonconstructive maximality arguments from commutative algebra
- PROGRAMS = state-based sequential algorithms

Our main technique for extraction will be:

Gödel's functional ("Dialectica") interpretation + **some cleverness**

Introduction

This talk is about the extraction of programs from proofs:

$$\text{PROOFS} \rightarrow \text{PROGRAMS}$$

More specifically, for us:

- PROOFS = nonconstructive maximality arguments from commutative algebra
- PROGRAMS = state-based sequential algorithms

Our main technique for extraction will be:

$$\text{Gödel's functional ("Dialectica") interpretation} + \textbf{some cleverness}$$

Introduction

This talk is about the extraction of programs from proofs:

$$\text{PROOFS} \to \text{PROGRAMS}$$

More specifically, for us:

- PROOFS = nonconstructive maximality arguments from commutative algebra
- PROGRAMS = state-based sequential algorithms

Our main technique for extraction will be:

Gödel's functional ("Dialectica") interpretation + **some cleverness**

Gödel's functional interpretation: What is it?

Formally speaking, a translation from formulas A is some logical theory $\mathcal{L}$ to formulas $\exists x \forall y A_D(x,y)$ in some (related) theory $\mathcal{P}$. Key points:

- $A_D(x,y)$ is 'computationally neutral',
- terms of $\mathcal{P}$ are usually those of some typed lambda calculus,
- for classical theories, we first apply a negative translation i.e. $A \mapsto A^N \mapsto \exists x \forall y A^N_D(x,y)$,
- key results are **soundness theorems**.

Theorem (Gödel - published 1958, already conceived 1930's)

Let A be a formula in the language of PA. *Then whenever* PA $\vdash A$, *there is some term t of System T such that $T \vdash \forall y A^N_D(t,y)$.*

Modern applications comprise:

1. Case studies which explore term extraction in different areas of mathematics,
2. New soundness theorems ('logical metatheorems') which describe general phenomena.

Gödel's functional interpretation: What is it?

Formally speaking, a translation from formulas A is some logical theory $\mathcal{L}$ to formulas $\exists x \forall y A_D(x,y)$ in some (related) theory $\mathcal{P}$. Key points:

- $A_D(x,y)$ is 'computationally neutral',
- terms of $\mathcal{P}$ are usually those of some typed lambda calculus,
- for classical theories, we first apply a negative translation i.e. $A \mapsto A^N \mapsto \exists x \forall y A_D^N(x,y)$,
- key results are **soundness theorems**.

Theorem (Gödel - published 1958, already conceived 1930's)

Let A be a formula in the language of PA. *Then whenever* PA $\vdash A$, *there is some term t of System T such that $T \vdash \forall y A_D^N(t,y)$.*

Modern applications comprise:

1. Case studies which explore term extraction in different areas of mathematics,
2. New soundness theorems ('logical metatheorems') which describe general phenomena.

Gödel's functional interpretation: What is it?

Formally speaking, a translation from formulas A is some logical theory $\mathcal{L}$ to formulas $\exists x \forall y A_D(x,y)$ in some (related) theory $\mathcal{P}$. Key points:

- $A_D(x,y)$ is 'computationally neutral',
- terms of $\mathcal{P}$ are usually those of some typed lambda calculus,
- for classical theories, we first apply a negative translation i.e. $A \mapsto A^N \mapsto \exists x \forall y A_D^N(x,y)$,
- key results are **soundness theorems**.

Theorem (Gödel - published 1958, already conceived 1930's)

Let A be a formula in the language of PA*. Then whenever* PA $\vdash A$*, there is some term t of System T such that $T \vdash \forall y A_D^N(t,y)$.*

Modern applications comprise:

1. Case studies which explore term extraction in different areas of mathematics,
2. New soundness theorems ('logical metatheorems') which describe general phenomena.

Gödel's functional interpretation: What is it?

Formally speaking, a translation from formulas A is some logical theory $\mathcal{L}$ to formulas $\exists x \forall y A_D(x,y)$ in some (related) theory $\mathcal{P}$. Key points:

- $A_D(x,y)$ is 'computationally neutral',
- terms of $\mathcal{P}$ are usually those of some typed lambda calculus,
- for classical theories, we first apply a negative translation i.e. $A \mapsto A^N \mapsto \exists x \forall y A^N_D(x,y)$,
- key results are **soundness theorems**.

Theorem (Gödel - published 1958, already conceived 1930's)

Let A be a formula in the language of PA*. Then whenever* PA $\vdash A$*, there is some term t of System T such that $T \vdash \forall y A^N_D(t,y)$.*

Modern applications comprise:

1. Case studies which explore term extraction in different areas of mathematics,
2. New soundness theorems ('logical metatheorems') which describe general phenomena.

Gödel's functional interpretation: What is it?

Formally speaking, a translation from formulas A is some logical theory $\mathcal{L}$ to formulas $\exists x \forall y A_D(x,y)$ in some (related) theory $\mathcal{P}$. Key points:

- $A_D(x,y)$ is 'computationally neutral',
- terms of $\mathcal{P}$ are usually those of some typed lambda calculus,
- for classical theories, we first apply a negative translation i.e. $A \mapsto A^N \mapsto \exists x \forall y A^N_D(x,y)$,
- key results are **soundness theorems**.

Theorem (Gödel - published 1958, already conceived 1930's)

Let A be a formula in the language of PA*. Then whenever* PA $\vdash A$*, there is some term t of System T such that* $T \vdash \forall y A^N_D(t,y)$.

Modern applications comprise:

1. Case studies which explore term extraction in different areas of mathematics,
2. New soundness theorems ('logical metatheorems') which describe general phenomena.

Gödel's functional interpretation: What is it?

Formally speaking, a translation from formulas A is some logical theory $\mathcal{L}$ to formulas $\exists x \forall y A_D(x, y)$ in some (related) theory $\mathcal{P}$. Key points:

- $A_D(x, y)$ is 'computationally neutral',
- terms of $\mathcal{P}$ are usually those of some typed lambda calculus,
- for classical theories, we first apply a negative translation i.e. $A \mapsto A^N \mapsto \exists x \forall y A^N_D(x, y)$,
- key results are **soundness theorems**.

Theorem (Gödel - published 1958, already conceived 1930's)

Let A be a formula in the language of PA. *Then whenever* PA $\vdash A$, *there is some term t of System T such that* T $\vdash \forall y A^N_D(t, y)$.

Modern applications comprise:

1. Case studies which explore term extraction in different areas of mathematics,
2. New soundness theorems ('logical metatheorems') which describe general phenomena.

Gödel's functional interpretation: What is it?

Formally speaking, a translation from formulas A is some logical theory $\mathcal{L}$ to formulas $\exists x \forall y A_D(x, y)$ in some (related) theory $\mathcal{P}$. Key points:

- $A_D(x, y)$ is 'computationally neutral',
- terms of $\mathcal{P}$ are usually those of some typed lambda calculus,
- for classical theories, we first apply a negative translation i.e. $A \mapsto A^N \mapsto \exists x \forall y A_D^N(x, y)$,
- key results are **soundness theorems**.

Theorem (Gödel - published 1958, already conceived 1930's)

Let A be a formula in the language of PA*. Then whenever* PA $\vdash A$*, there is some term t of System T such that $T \vdash \forall y A_D^N(t, y)$.*

Modern applications comprise:

1. Case studies which explore term extraction in different areas of mathematics,
2. New soundness theorems ('logical metatheorems') which describe general phenomena.

Gödel's functional interpretation: What is it?

Formally speaking, a translation from formulas A is some logical theory $\mathcal{L}$ to formulas $\exists x \forall y A_D(x, y)$ in some (related) theory $\mathcal{P}$. Key points:

- $A_D(x, y)$ is 'computationally neutral',
- terms of $\mathcal{P}$ are usually those of some typed lambda calculus,
- for classical theories, we first apply a negative translation i.e. $A \mapsto A^N \mapsto \exists x \forall y A_D^N(x, y)$,
- key results are **soundness theorems**.

Theorem (Gödel - published 1958, already conceived 1930's)

Let A be a formula in the language of PA. *Then whenever* PA $\vdash A$*, there is some term t of System T such that $T \vdash \forall y A_D^N(t, y)$.*

Modern applications comprise:

1. Case studies which explore term extraction in different areas of mathematics,
2. New soundness theorems ('logical metatheorems') which describe general phenomena.

How does the functional interpretation deal with $\exists\forall$ theorems?

Many non-constructive theorems have the form $\exists x \forall y P(x,y)$ for decidable $P(x,y)$.

Example (Drinkers paradox: classical variant)

$\exists x \forall y (\neg D(x) \vee D(y))$ for $D(z)$ decidable. A witness for x not computable in general.

$$\begin{aligned} \exists x \forall y \; P(x,y) &\mapsto \neg\neg\exists x \forall y P(x,y) \\ &\mapsto \neg\forall x \exists y \neg P(x,y) \\ &\mapsto \neg\exists\phi\forall x \neg P(x,\phi x) \\ &\mapsto \forall\phi\exists x P(x,\phi x) \\ &\mapsto \exists F \forall\phi P(F\phi, \phi(F\phi)). \end{aligned}$$

Example (Drinkers paradox: constructive variant)

$\exists F \forall \phi (\neg D(F\phi) \vee D(\phi(F\phi)))$. Can be solved by setting

$$F\phi := \begin{cases} 0 & \text{if } D(\phi 0) \\ \phi 0 & \text{if } \neg D(\phi 0) \end{cases}$$

How does the functional interpretation deal with $\exists\forall$ theorems?

Many non-constructive theorems have the form $\exists x \forall y P(x,y)$ for decidable $P(x,y)$.

Example (Drinkers paradox: classical variant)

$\exists x \forall y(\neg D(x) \vee D(y))$ for $D(z)$ decidable. A witness for x not computable in general.

$$\begin{aligned}
\exists x \forall y\; P(x,y) &\mapsto \neg\neg\exists x \forall y P(x,y) \\
&\mapsto \neg\forall x \exists y \neg P(x,y) \\
&\mapsto \neg\exists \phi \forall x \neg P(x, \phi x) \\
&\mapsto \forall \phi \exists x P(x, \phi x) \\
&\mapsto \exists F \forall \phi P(F\phi, \phi(F\phi)).
\end{aligned}$$

Example (Drinkers paradox: constructive variant)

$\exists F \forall \phi(\neg D(F\phi) \vee D(\phi(F\phi)))$. Can be solved by setting

$$F\phi := \begin{cases} 0 & \text{if } D(\phi 0) \\ \phi 0 & \text{if } \neg D(\phi 0) \end{cases}$$

How does the functional interpretation deal with $\exists\forall$ theorems?

Many non-constructive theorems have the form $\exists x \forall y P(x,y)$ for decidable $P(x,y)$.

Example (Drinkers paradox: classical variant)

$\exists x \forall y(\neg D(x) \vee D(y))$ for $D(z)$ decidable. A witness for x not computable in general.

$$\begin{aligned} \exists x \forall y\; P(x,y) &\mapsto \neg\neg\exists x\forall y P(x,y) \\ &\mapsto \neg\forall x \exists y \neg P(x,y) \\ &\mapsto \neg\exists\phi\forall x\neg P(x,\phi x) \\ &\mapsto \forall\phi\exists x P(x,\phi x) \\ &\mapsto \exists F\forall\phi P(F\phi,\phi(F\phi)). \end{aligned}$$

Example (Drinkers paradox: constructive variant)

$\exists F\forall\phi(\neg D(F\phi) \vee D(\phi(F\phi)))$. Can be solved by setting

$$F\phi := \begin{cases} 0 & \text{if } D(\phi 0) \\ \phi 0 & \text{if } \neg D(\phi 0) \end{cases}$$

How does the functional interpretation deal with $\exists\forall$ theorems?

Many non-constructive theorems have the form $\exists x \forall y P(x,y)$ for decidable $P(x,y)$.

Example (Drinkers paradox: classical variant)

$\exists x \forall y (\neg D(x) \vee D(y))$ for $D(z)$ decidable. A witness for x not computable in general.

$$\begin{aligned}
\exists x \forall y\ P(x,y) &\mapsto \neg\neg\exists x \forall y P(x,y) \\
&\mapsto \neg\forall x \exists y \neg P(x,y) \\
&\mapsto \neg\exists\phi\forall x \neg P(x,\phi x) \\
&\mapsto \forall\phi\exists x P(x,\phi x) \\
&\mapsto \exists F \forall\phi P(F\phi,\phi(F\phi)).
\end{aligned}$$

Example (Drinkers paradox: constructive variant)

$\exists F \forall \phi (\neg D(F\phi) \vee D(\phi(F\phi)))$. Can be solved by setting

$$F\phi := \begin{cases} 0 & \text{if } D(\phi 0) \\ \phi 0 & \text{if } \neg D(\phi 0) \end{cases}$$

How does the functional interpretation deal with ∃∀ theorems?

Many non-constructive theorems have the form $\exists x \forall y P(x,y)$ for decidable $P(x,y)$.

Example (Drinkers paradox: classical variant)

$\exists x \forall y (\neg D(x) \vee D(y))$ for $D(z)$ decidable. A witness for x not computable in general.

$$\begin{aligned}
\exists x \forall y\ P(x,y) &\mapsto \neg\neg\exists x \forall y P(x,y)\\
&\mapsto \neg\forall x \exists y \neg P(x,y)\\
&\mapsto \neg\exists \phi \forall x \neg P(x,\phi x)\\
&\mapsto \forall \phi \exists x P(x,\phi x)\\
&\mapsto \exists F \forall \phi P(F\phi,\phi(F\phi)).
\end{aligned}$$

Example (Drinkers paradox: constructive variant)

$\exists F \forall \phi (\neg D(F\phi) \vee D(\phi(F\phi)))$. Can be solved by setting

$$F\phi := \begin{cases} 0 & \text{if } D(\phi 0) \\ \phi 0 & \text{if } \neg D(\phi 0) \end{cases}$$

How does the functional interpretation deal with ∃∀ theorems?

Many non-constructive theorems have the form $\exists x \forall y P(x,y)$ for decidable $P(x,y)$.

Example (Drinkers paradox: classical variant)

$\exists x \forall y(\neg D(x) \vee D(y))$ for $D(z)$ decidable. A witness for x not computable in general.

$$\begin{aligned} \exists x \forall y\ P(x,y) &\mapsto \neg\neg\exists x \forall y P(x,y) \\ &\mapsto \neg\forall x \exists y \neg P(x,y) \\ &\mapsto \neg\exists\phi\forall x \neg P(x,\phi x) \\ &\mapsto \forall\phi\exists x P(x,\phi x) \\ &\mapsto \exists F\forall\phi P(F\phi,\phi(F\phi)). \end{aligned}$$

Example (Drinkers paradox: constructive variant)

$\exists F \forall \phi(\neg D(F\phi) \vee D(\phi(F\phi))$. Can be solved by setting

$$F\phi := \begin{cases} 0 & \text{if } D(\phi 0) \\ \phi 0 & \text{if } \neg D(\phi 0) \end{cases}$$

How does the functional interpretation deal with $\exists\forall$ theorems?

Many non-constructive theorems have the form $\exists x \forall y P(x,y)$ for decidable $P(x,y)$.

Example (Drinkers paradox: classical variant)

$\exists x \forall y(\neg D(x) \vee D(y))$ for $D(z)$ decidable. A witness for x not computable in general.

$$\begin{aligned}
\exists x \forall y\ P(x,y) &\mapsto \neg\neg\exists x \forall y P(x,y) \\
&\mapsto \neg\forall x \exists y \neg P(x,y) \\
&\mapsto \neg\exists \phi \forall x \neg P(x, \phi x) \\
&\mapsto \forall \phi \exists x P(x, \phi x) \\
&\mapsto \exists F \forall \phi P(F\phi, \phi(F\phi)).
\end{aligned}$$

Example (Drinkers paradox: constructive variant)

$\exists F \forall \phi(\neg D(F\phi) \vee D(\phi(F\phi))$. Can be solved by setting

$$F\phi := \begin{cases} 0 & \text{if } D(\phi 0) \\ \phi 0 & \text{if } \neg D(\phi 0) \end{cases}$$

How does the functional interpretation deal with ∃∀ theorems?

Many non-constructive theorems have the form $\exists x \forall y P(x,y)$ for decidable $P(x,y)$.

Example (Drinkers paradox: classical variant)

$\exists x \forall y (\neg D(x) \vee D(y))$ for $D(z)$ decidable. A witness for x not computable in general.

$$\begin{aligned}
\exists x \forall y\ P(x,y) &\mapsto \neg\neg\exists x \forall y P(x,y) \\
&\mapsto \neg\forall x \exists y \neg P(x,y) \\
&\mapsto \neg\exists \phi \forall x \neg P(x, \phi x) \\
&\mapsto \forall \phi \exists x P(x, \phi x) \\
&\mapsto \exists F \forall \phi P(F\phi, \phi(F\phi)).
\end{aligned}$$

Example (Drinkers paradox: constructive variant)

$\exists F \forall \phi (\neg D(F\phi) \vee D(\phi(F\phi))$. Can be solved by setting

$$F\phi := \begin{cases} 0 & \text{if } D(\phi 0) \\ \phi 0 & \text{if } \neg D(\phi 0) \end{cases}$$

How does the functional interpretation deal with $\exists\forall$ theorems?

Many non-constructive theorems have the form $\exists x \forall y P(x,y)$ for decidable $P(x,y)$.

Example (Drinkers paradox: classical variant)

$\exists x \forall y (\neg D(x) \vee D(y))$ for $D(z)$ decidable. A witness for x not computable in general.

$$\begin{aligned} \exists x \forall y \; P(x,y) &\mapsto \neg\neg\exists x \forall y P(x,y) \\ &\mapsto \neg\forall x \exists y \neg P(x,y) \\ &\mapsto \neg\exists\phi\forall x \neg P(x,\phi x) \\ &\mapsto \forall\phi\exists x P(x,\phi x) \\ &\mapsto \exists F \forall\phi P(F\phi, \phi(F\phi)). \end{aligned}$$

Example (Drinkers paradox: constructive variant)

$\exists F \forall \phi (\neg D(F\phi) \vee D(\phi(F\phi)))$. Can be solved by setting

$$F\phi := \begin{cases} 0 & \text{if } D(\phi 0) \\ \phi 0 & \text{if } \neg D(\phi 0) \end{cases}$$

How does the functional interpretation deal with ∃∀ theorems?

Many non-constructive theorems have the form $\exists x \forall y P(x,y)$ for decidable $P(x,y)$.

Example (Drinkers paradox: classical variant)

$\exists x \forall y(\neg D(x) \vee D(y))$ for $D(z)$ decidable. A witness for x not computable in general.

$$\begin{aligned}
\exists x \forall y\ P(x,y) &\mapsto \neg\neg\exists x \forall y P(x,y) \\
&\mapsto \neg\forall x \exists y \neg P(x,y) \\
&\mapsto \neg\exists\phi\forall x\neg P(x,\phi x) \\
&\mapsto \forall\phi\exists x P(x,\phi x) \\
&\mapsto \exists F\forall\phi P(F\phi,\phi(F\phi)).
\end{aligned}$$

Example (Drinkers paradox: constructive variant)

$\exists F \forall \phi(\neg D(F\phi) \vee D(\phi(F\phi)))$. Can be solved by setting

$$F\phi := \begin{cases} 0 & \text{if } D(\phi 0) \\ \phi 0 & \text{if } \neg D(\phi 0) \end{cases}$$

What if a ∃∀ statement is used as a *lemma* in the proof of a ∃ theorem?

$$\begin{aligned}(\exists x \forall y P(x,y) \to \exists v Q(v)) &\mapsto \forall x \exists y, v (P(x,y) \to Q(v)) \\ &\mapsto \exists g, h \forall x (P(x, gx) \to Q(hx))\end{aligned}$$

Example (Drinkers paradox as a lemma)

$\exists x \forall y (\neg D(x) \vee D(y)) \to \exists v (\neg D(v+2) \vee D(3v+1))$ is valid, and would be translated to

$$\exists g, h \forall x (\neg D(x) \vee D(gx) \to \neg D(hx+2) \vee D(3hx+1)).$$

Solved by setting $gx := 3x - 5$ and $hx := x - 2$.

$$\frac{\exists x \forall y P(x,y) \quad \exists x \forall y P(x,y) \to \exists v Q(v)}{\exists v Q(v)} \mapsto \frac{\forall \phi P(F\phi, \phi(F\phi)) \quad \forall x (P(x, gx) \to Q(hx))}{Q(-)}$$

$$\mapsto \frac{P(Fg, g(Fg)) \quad P(Fg, g(Fg)) \to Q(h(Fg))}{Q(h(Fg))}$$

What if a $\exists\forall$ statement is used as a *lemma* in the proof of a $\exists$ theorem?

$$\begin{aligned}(\exists x \forall y P(x,y) \to \exists v Q(v)) &\mapsto \forall x \exists y, v(P(x,y) \to Q(v)) \\ &\mapsto \exists g, h \forall x (P(x, gx) \to Q(hx))\end{aligned}$$

Example (Drinkers paradox as a lemma)

$\exists x \forall y(\neg D(x) \vee D(y)) \to \exists v(\neg D(v+2) \vee D(3v+1))$ is valid, and would be translated to

$$\exists g, h \forall x(\neg D(x) \vee D(gx) \to \neg D(hx+2) \vee D(3hx+1)).$$

Solved by setting $gx := 3x - 5$ and $hx := x - 2$.

$$\frac{\exists x \forall y P(x,y) \quad \exists x \forall y P(x,y) \to \exists v Q(v)}{\exists v Q(v)} \mapsto \frac{\forall \phi P(F\phi, \phi(F\phi)) \quad \forall x (P(x, gx) \to Q(hx))}{Q(-)}$$

$$\mapsto \frac{P(Fg, g(Fg)) \quad P(Fg, g(Fg)) \to Q(h(Fg))}{Q(h(Fg))}$$

What if a ∃∀ statement is used as a *lemma* in the proof of a ∃ theorem?

$$(\exists x \forall y P(x,y) \to \exists v Q(v)) \mapsto \forall x \exists y, v(P(x,y) \to Q(v))$$

$$\mapsto \exists g, h \forall x(P(x,gx) \to Q(hx))$$

Example (Drinkers paradox as a lemma)

$\exists x \forall y(\neg D(x) \vee D(y)) \to \exists v(\neg D(v+2) \vee D(3v+1))$ is valid, and would be translated to

$$\exists g, h \forall x(\neg D(x) \vee D(gx) \to \neg D(hx+2) \vee D(3hx+1)).$$

Solved by setting $gx := 3x - 5$ and $hx := x - 2$.

$$\frac{\exists x \forall y P(x,y) \quad \exists x \forall y P(x,y) \to \exists v Q(v)}{\exists v Q(v)} \mapsto \frac{\forall \phi P(F\phi, \phi(F\phi)) \quad \forall x(P(x,gx) \to Q(hx))}{Q(-)}$$

$$\mapsto \frac{P(Fg, g(Fg)) \quad P(Fg, g(Fg)) \to Q(h(Fg))}{Q(h(Fg))}$$

What if a ∃∀ statement is used as a *lemma* in the proof of a ∃ theorem?

$$\begin{aligned}(\exists x \forall y P(x,y) \to \exists v Q(v)) &\mapsto \forall x \exists y, v (P(x,y) \to Q(v)) \\ &\mapsto \exists g, h \forall x (P(x, gx) \to Q(hx))\end{aligned}$$

Example (Drinkers paradox as a lemma)

$\exists x \forall y (\neg D(x) \vee D(y)) \to \exists v (\neg D(v+2) \vee D(3v+1))$ is valid, and would be translated to

$$\exists g, h \forall x (\neg D(x) \vee D(gx) \to \neg D(hx+2) \vee D(3hx+1)).$$

Solved by setting $gx := 3x - 5$ and $hx := x - 2$.

$$\frac{\exists x \forall y P(x,y) \quad \exists x \forall y P(x,y) \to \exists v Q(v)}{\exists v Q(v)} \mapsto \frac{\forall \phi P(F\phi, \phi(F\phi)) \quad \forall x (P(x, gx) \to Q(hx))}{Q(-)}$$

$$\mapsto \frac{P(Fg, g(Fg)) \quad P(Fg, g(Fg)) \to Q(h(Fg))}{Q(h(Fg))}$$

What if a ∃∀ statement is used as a *lemma* in the proof of a ∃ theorem?

$$\begin{aligned}(\exists x \forall y P(x,y) \to \exists v Q(v)) &\mapsto \forall x \exists y, v (P(x,y) \to Q(v)) \\ &\mapsto \exists g, h \forall x (P(x, gx) \to Q(hx))\end{aligned}$$

Example (Drinkers paradox as a lemma)

$\exists x \forall y (\neg D(x) \vee D(y)) \to \exists v (\neg D(v+2) \vee D(3v+1))$ is valid, and would be translated to

$$\exists g, h \forall x (\neg D(x) \vee D(gx) \to \neg D(hx+2) \vee D(3hx+1)).$$

Solved by setting $gx := 3x - 5$ and $hx := x - 2$.

$$\begin{aligned}\frac{\exists x \forall y P(x,y) \quad \exists x \forall y P(x,y) \to \exists v Q(v)}{\exists v Q(v)} &\mapsto \frac{\forall \phi P(F\phi, \phi(F\phi)) \quad \forall x (P(x, gx) \to Q(hx))}{Q(-)} \\ &\mapsto \frac{P(Fg, g(Fg)) \quad P(Fg, g(Fg)) \to Q(h(Fg))}{Q(h(Fg))}\end{aligned}$$

What if a ∃∀ statement is used as a *lemma* in the proof of a ∃ theorem?

$$\begin{aligned}(\exists x\forall y P(x,y)\to\exists v Q(v)) &\mapsto \forall x\exists y,v(P(x,y)\to Q(v))\\ &\mapsto \exists g,h\forall x(P(x,gx)\to Q(hx))\end{aligned}$$

Example (Drinkers paradox as a lemma)

$\exists x\forall y(\neg D(x)\vee D(y))\to\exists v(\neg D(v+2)\vee D(3v+1))$ is valid, and would be translated to

$$\exists g,h\forall x(\neg D(x)\vee D(gx)\to\neg D(hx+2)\vee D(3hx+1)).$$

Solved by setting $gx := 3x-5$ and $hx := x-2$.

$$\frac{\exists x\forall y P(x,y) \quad \exists x\forall y P(x,y)\to\exists v Q(v)}{\exists v Q(v)} \mapsto \frac{\forall\phi P(F\phi,\phi(F\phi)) \quad \forall x(P(x,gx)\to Q(hx))}{Q(-)}$$

$$\mapsto \frac{P(Fg,g(Fg)) \quad P(Fg,g(Fg))\to Q(h(Fg))}{Q(h(Fg))}$$

What if a ∃∀ statement is used as a *lemma* in the proof of a ∃ theorem?

$$(\exists x\forall y P(x,y) \to \exists v Q(v)) \mapsto \forall x\exists y, v(P(x,y) \to Q(v))$$

$$\mapsto \exists g, h\forall x(P(x,gx) \to Q(hx))$$

Example (Drinkers paradox as a lemma)

$\exists x\forall y(\neg D(x) \vee D(y)) \to \exists v(\neg D(v+2) \vee D(3v+1))$ is valid, and would be translated to

$$\exists g, h\forall x(\neg D(x) \vee D(gx) \to \neg D(hx+2) \vee D(3hx+1)).$$

Solved by setting $gx := 3x - 5$ and $hx := x - 2$.

$$\frac{\exists x\forall y P(x,y) \quad \exists x\forall y P(x,y) \to \exists v Q(v)}{\exists v Q(v)} \mapsto \frac{\forall \phi P(F\phi, \phi(F\phi)) \quad \forall x(P(x,gx) \to Q(hx))}{Q(-)}$$

$$\mapsto \frac{P(Fg, g(Fg)) \quad P(Fg, g(Fg)) \to Q(h(Fg))}{Q(h(Fg))}$$

What if a $\exists\forall$ statement is used as a *lemma* in the proof of a $\exists$ theorem?

$$\begin{aligned}(\exists x\forall y P(x,y) \to \exists v Q(v)) &\mapsto \forall x\exists y, v(P(x,y) \to Q(v))\\ &\mapsto \exists g, h\forall x(P(x,gx) \to Q(hx))\end{aligned}$$

Example (Drinkers paradox as a lemma)

$\exists x\forall y(\neg D(x) \vee D(y)) \to \exists v(\neg D(v+2) \vee D(3v+1))$ is valid, and would be translated to

$$\exists g, h\forall x(\neg D(x) \vee D(gx) \to \neg D(hx+2) \vee D(3hx+1)).$$

Solved by setting $gx := 3x - 5$ and $hx := x - 2$.

$$\frac{\exists x\forall y P(x,y) \quad \exists x\forall y P(x,y) \to \exists v Q(v)}{\exists v Q(v)} \mapsto \frac{\forall\phi P(F\phi, \phi(F\phi)) \quad \forall x(P(x,gx) \to Q(hx))}{Q(-)}$$

$$\mapsto \frac{P(Fg, g(Fg)) \quad P(Fg, g(Fg)) \to Q(h(Fg))}{Q(h(Fg))}$$

Direct witnesses from nonconstructive proofs

Example (A nonconstructive proof of $\exists v(\neg D(v+2) \vee D(3v+1))$)

$$\frac{\exists x \forall y(\neg D(x) \vee D(y)) \quad \exists x \forall y(\neg D(x) \vee D(y)) \to \exists v(\neg D(v+2) \vee D(3v+1))}{\exists v(\neg D(v+2) \vee D(3v+1))}$$

Define $F\phi := \begin{cases} 0 & \text{if } D(\phi 0) \\ \phi 0 & \text{if } \neg D(\phi 0) \end{cases}$ and $gx := 3x - 5$ and $hx := x - 2$. Then:

$$\neg D(Fg) \vee D(g(Fg)) \quad \text{and} \quad \neg D(Fg) \vee D(g(Fg)) \to \neg D(h(Fg) + 2) \vee D(3h(Fg) + 1)$$

Therefore $\exists v(\neg D(v+2) \vee D(3v+1))$ is witnessed by $v := h(Fg)$ i.e.

$$v := \begin{cases} 0 - 2 & \text{if } D(3 \cdot 0 - 5) \\ (3 \cdot 0 - 5) - 2 & \text{if } \neg D(3 \cdot 0 - 5) \end{cases} = \begin{cases} -2 & \text{if } D(-5) \\ -7 & \text{if } \neg D(-5) \end{cases}$$

We can check this directly: If $D(-5)$ then $\neg D(-2+2) \vee D(3 \cdot (-2) + 1)$, otherwise if $\neg D(-5)$ then $\neg D(-7+2) \vee D(3 \cdot (-7) + 1)$.

Direct witnesses from nonconstructive proofs

Example (A nonconstructive proof of $\exists v(\neg D(v+2) \vee D(3v+1))$)

$$\frac{\exists x \forall y(\neg D(x) \vee D(y)) \quad \exists x \forall y(\neg D(x) \vee D(y)) \to \exists v(\neg D(v+2) \vee D(3v+1))}{\exists v(\neg D(v+2) \vee D(3v+1))}$$

Define $F\phi := \begin{cases} 0 & \text{if } D(\phi 0) \\ \phi 0 & \text{if } \neg D(\phi 0) \end{cases}$ and $gx := 3x - 5$ and $hx := x - 2$. Then:

$$\neg D(Fg) \vee D(g(Fg)) \quad \text{and} \quad \neg D(Fg) \vee D(g(Fg)) \to \neg D(h(Fg)+2) \vee D(3h(Fg)+1)$$

Therefore $\exists v(\neg D(v+2) \vee D(3v+1))$ is witnessed by $v := h(Fg)$ i.e.

$$v := \begin{cases} 0 - 2 & \text{if } D(3 \cdot 0 - 5) \\ (3 \cdot 0 - 5) - 2 & \text{if } \neg D(3 \cdot 0 - 5) \end{cases} = \begin{cases} -2 & \text{if } D(-5) \\ -7 & \text{if } \neg D(-5) \end{cases}$$

We can check this directly: If $D(-5)$ then $\neg D(-2+2) \vee D(3 \cdot (-2) + 1)$, otherwise if $\neg D(-5)$ then $\neg D(-7+2) \vee D(3 \cdot (-7) + 1)$.

Direct witnesses from nonconstructive proofs

Example (A nonconstructive proof of $\exists v(\neg D(v+2) \vee D(3v+1))$)

$$\frac{\exists x \forall y(\neg D(x) \vee D(y)) \quad \exists x \forall y(\neg D(x) \vee D(y)) \to \exists v(\neg D(v+2) \vee D(3v+1))}{\exists v(\neg D(v+2) \vee D(3v+1))}$$

Define $F\phi := \begin{cases} 0 & \text{if } D(\phi 0) \\ \phi 0 & \text{if } \neg D(\phi 0) \end{cases}$ and $gx := 3x - 5$ and $hx := x - 2$. Then:

$$\neg D(Fg) \vee D(g(Fg)) \quad \text{and} \quad \neg D(Fg) \vee D(g(Fg)) \to \neg D(h(Fg)+2) \vee D(3h(Fg)+1)$$

Therefore $\exists v(\neg D(v+2) \vee D(3v+1))$ is witnessed by $v := h(Fg)$ i.e.

$$v := \begin{cases} 0 - 2 & \text{if } D(3 \cdot 0 - 5) \\ (3 \cdot 0 - 5) - 2 & \text{if } \neg D(3 \cdot 0 - 5) \end{cases} = \begin{cases} -2 & \text{if } D(-5) \\ -7 & \text{if } \neg D(-5) \end{cases}$$

We can check this directly: If $D(-5)$ then $\neg D(-2+2) \vee D(3 \cdot (-2) + 1)$, otherwise if $\neg D(-5)$ then $\neg D(-7+2) \vee D(3 \cdot (-7) + 1)$.

Direct witnesses from nonconstructive proofs

Example (A nonconstructive proof of $\exists v(\neg D(v+2) \vee D(3v+1))$)

$$\frac{\exists x \forall y(\neg D(x) \vee D(y)) \quad \exists x \forall y(\neg D(x) \vee D(y)) \to \exists v(\neg D(v+2) \vee D(3v+1))}{\exists v(\neg D(v+2) \vee D(3v+1))}$$

Define $F\phi := \begin{cases} 0 & \text{if } D(\phi 0) \\ \phi 0 & \text{if } \neg D(\phi 0) \end{cases}$ and $gx := 3x - 5$ and $hx := x - 2$. Then:

$$\neg D(Fg) \vee D(g(Fg)) \quad \text{and} \quad \neg D(Fg) \vee D(g(Fg)) \to \neg D(h(Fg)+2) \vee D(3h(Fg)+1)$$

Therefore $\exists v(\neg D(v+2) \vee D(3v+1))$ is witnessed by $v := h(Fg)$ i.e.

$$v := \begin{cases} 0 - 2 & \text{if } D(3 \cdot 0 - 5) \\ (3 \cdot 0 - 5) - 2 & \text{if } \neg D(3 \cdot 0 - 5) \end{cases} = \begin{cases} -2 & \text{if } D(-5) \\ -7 & \text{if } \neg D(-5) \end{cases}$$

We can check this directly: If $D(-5)$ then $\neg D(-2+2) \vee D(3 \cdot (-2)+1)$, otherwise if $\neg D(-5)$ then $\neg D(-7+2) \vee D(3 \cdot (-7)+1)$.

Direct witnesses from nonconstructive proofs

Example (A nonconstructive proof of $\exists v(\neg D(v+2) \vee D(3v+1))$)

$$\frac{\exists x \forall y(\neg D(x) \vee D(y)) \quad \exists x \forall y(\neg D(x) \vee D(y)) \to \exists v(\neg D(v+2) \vee D(3v+1))}{\exists v(\neg D(v+2) \vee D(3v+1))}$$

Define $F\phi := \begin{cases} 0 & \text{if } D(\phi 0) \\ \phi 0 & \text{if } \neg D(\phi 0) \end{cases}$ and $gx := 3x - 5$ and $hx := x - 2$. Then:

$$\neg D(Fg) \vee D(g(Fg)) \ \text{ and } \ \neg D(Fg) \vee D(g(Fg)) \to \neg D(h(Fg)+2) \vee D(3h(Fg)+1)$$

Therefore $\exists v(\neg D(v+2) \vee D(3v+1))$ is witnessed by $v := h(Fg)$ i.e.

$$v := \begin{cases} 0 - 2 & \text{if } D(3 \cdot 0 - 5) \\ (3 \cdot 0 - 5) - 2 & \text{if } \neg D(3 \cdot 0 - 5) \end{cases} = \begin{cases} -2 & \text{if } D(-5) \\ -7 & \text{if } \neg D(-5) \end{cases}$$

We can check this directly: If $D(-5)$ then $\neg D(-2+2) \vee D(3 \cdot (-2)+1)$, otherwise if $\neg D(-5)$ then $\neg D(-7+2) \vee D(3 \cdot (-7)+1)$.

Direct witnesses from nonconstructive proofs

Example (A nonconstructive proof of $\exists v(\neg D(v+2) \vee D(3v+1))$)

$$\frac{\exists x \forall y(\neg D(x) \vee D(y)) \quad \exists x \forall y(\neg D(x) \vee D(y)) \to \exists v(\neg D(v+2) \vee D(3v+1))}{\exists v(\neg D(v+2) \vee D(3v+1))}$$

Define $F\phi := \begin{cases} 0 & \text{if } D(\phi 0) \\ \phi 0 & \text{if } \neg D(\phi 0) \end{cases}$ and $gx := 3x - 5$ and $hx := x - 2$. Then:

$$\neg D(Fg) \vee D(g(Fg)) \quad \text{and} \quad \neg D(Fg) \vee D(g(Fg)) \to \neg D(h(Fg)+2) \vee D(3h(Fg)+1)$$

Therefore $\exists v(\neg D(v+2) \vee D(3v+1))$ is witnessed by $v := h(Fg)$ i.e.

$$v := \begin{cases} 0 - 2 & \text{if } D(3 \cdot 0 - 5) \\ (3 \cdot 0 - 5) - 2 & \text{if } \neg D(3 \cdot 0 - 5) \end{cases} = \begin{cases} -2 & \text{if } D(-5) \\ -7 & \text{if } \neg D(-5) \end{cases}$$

We can check this directly: If $D(-5)$ then $\neg D(-2+2) \vee D(3 \cdot (-2) + 1)$, otherwise if $\neg D(-5)$ then $\neg D(-7+2) \vee D(3 \cdot (-7) + 1)$.

Direct witnesses from nonconstructive proofs

Example (A nonconstructive proof of $\exists v(\neg D(v+2) \vee D(3v+1))$)

$$\frac{\exists x \forall y(\neg D(x) \vee D(y)) \quad \exists x \forall y(\neg D(x) \vee D(y)) \to \exists v(\neg D(v+2) \vee D(3v+1))}{\exists v(\neg D(v+2) \vee D(3v+1))}$$

Define $F\phi := \begin{cases} 0 & \text{if } D(\phi 0) \\ \phi 0 & \text{if } \neg D(\phi 0) \end{cases}$ and $gx := 3x - 5$ and $hx := x - 2$. Then:

$$\neg D(Fg) \vee D(g(Fg)) \quad \text{and} \quad \neg D(Fg) \vee D(g(Fg)) \to \neg D(h(Fg)+2) \vee D(3h(Fg)+1)$$

Therefore $\exists v(\neg D(v+2) \vee D(3v+1))$ is witnessed by $v := h(Fg)$ i.e.

$$v := \begin{cases} 0 - 2 & \text{if } D(3 \cdot 0 - 5) \\ (3 \cdot 0 - 5) - 2 & \text{if } \neg D(3 \cdot 0 - 5) \end{cases} = \begin{cases} -2 & \text{if } D(-5) \\ -7 & \text{if } \neg D(-5) \end{cases}$$

We can check this directly: If $D(-5)$ then $\neg D(-2+2) \vee D(3 \cdot (-2) + 1)$, otherwise if $\neg D(-5)$ then $\neg D(-7+2) \vee D(3 \cdot (-7) + 1)$.

End of the logical part! Now to the world of computable algebra...

We now focus on 'textbook' proofs in commutative algebra which follow the above pattern. More specifically:

$\exists x \forall y P(x, y)$ $\sim$ a maximality principle (e.g. *'the ring R contains maximal ideal'*)

$\exists v Q(v)$ $\sim$ an existential theorem (e.g. *'the element r is nilpotent'*)

$\forall \phi \exists x P(x, \phi x)$ $\sim$ existence of 'approximately maximal' objects

F satisfying $\forall \phi P(r, \phi r)$ $\sim$ a sequential *bar recursive* algorithm

In this talk I will just sketch just one simple example, but the above scheme is much more general...

End of the logical part! Now to the world of computable algebra...

We now focus on ‘textbook’ proofs in commutative algebra which follow the above pattern. More specifically:

$\exists x \forall y P(x, y) \sim$ a maximality principle (e.g. *‘the ring R contains maximal ideal’*)

$\exists v Q(v) \sim$ an existential theorem (e.g. *‘the element r is nilpotent’*)

$\forall \phi \exists x P(x, \phi x) \sim$ existence of ‘approximately maximal’ objects

F satisfying $\forall \phi P(r, \phi r) \sim$ a sequential *bar recursive* algorithm

In this talk I will just sketch just one simple example, but the above scheme is much more general...

End of the logical part! Now to the world of computable algebra...

We now focus on ‘textbook’ proofs in commutative algebra which follow the above pattern. More specifically:

$\exists x \forall y P(x,y) \sim$ a maximality principle (e.g. *‘the ring R contains maximal ideal’*)

$\exists v Q(v) \sim$ an existential theorem (e.g. *‘the element r is nilpotent’*)

$\forall \phi \exists x P(x, \phi x) \sim$ existence of ‘approximately maximal’ objects

F satisfying $\forall \phi P(r, \phi r) \sim$ a sequential *bar recursive* algorithm

In this talk I will just sketch just one simple example, but the above scheme is much more general...

End of the logical part! Now to the world of computable algebra...

We now focus on 'textbook' proofs in commutative algebra which follow the above pattern. More specifically:

$$\exists x \forall y P(x,y) \sim \text{a maximality principle (e.g. \textit{'the ring R contains maximal ideal'})}$$

$$\exists v Q(v) \sim \text{an existential theorem (e.g. \textit{'the element r is nilpotent'})}$$

$$\forall \phi \exists x P(x, \phi x) \sim \text{existence of 'approximately maximal' objects}$$

$$F \text{ satisfying } \forall \phi P(r, \phi r) \sim \text{a sequential \textit{bar recursive} algorithm}$$

In this talk I will just sketch just one simple example, but the above scheme is much more general...

End of the logical part! Now to the world of computable algebra...

We now focus on 'textbook' proofs in commutative algebra which follow the above pattern. More specifically:

$\exists x \forall y P(x, y) \sim$ a maximality principle (e.g. *'the ring R contains maximal ideal'*)

$\exists v Q(v) \sim$ an existential theorem (e.g. *'the element r is nilpotent'*)

$\forall \phi \exists x P(x, \phi x) \sim$ existence of 'approximately maximal' objects

F satisfying $\forall \phi P(r, \phi r) \sim$ a sequential *bar recursive* algorithm

In this talk I will just sketch just one simple example, but the above scheme is much more general...

End of the logical part! Now to the world of computable algebra...

We now focus on 'textbook' proofs in commutative algebra which follow the above pattern. More specifically:

$$\begin{aligned} \exists x \forall y P(x,y) &\sim \text{a maximality principle (e.g. \textit{'the ring R contains maximal ideal'})} \\ \exists v Q(v) &\sim \text{an existential theorem (e.g. \textit{'the element r is nilpotent'})} \\ \forall \phi \exists x P(x, \phi x) &\sim \text{existence of 'approximately maximal' objects} \\ F \text{ satisfying } \forall \phi P(r, \phi r) &\sim \text{a sequential \textit{bar recursive} algorithm} \end{aligned}$$

In this talk I will just sketch just one simple example, but the above scheme is much more general...

End of the logical part! Now to the world of computable algebra...

We now focus on 'textbook' proofs in commutative algebra which follow the above pattern. More specifically:

$$\begin{aligned}
\exists x \forall y P(x,y) &\sim \text{a maximality principle (e.g. \textit{'the ring R contains maximal ideal'})}\\
\exists v Q(v) &\sim \text{an existential theorem (e.g. \textit{'the element r is nilpotent'})}\\
\forall \phi \exists x P(x, \phi x) &\sim \text{existence of 'approximately maximal' objects}\\
F \text{ satisfying } \forall \phi P(r, \phi r) &\sim \text{a sequential \textit{bar recursive} algorithm}
\end{aligned}$$

In this talk I will just sketch just one simple example, but the above scheme is much more general...

Some basic definitions

We start of with an abstract generating relation $\rhd$:

- X is a set.
- $\rhd$ is a relation on $\mathcal{P}_{fin}(X) \times X$, where we say '$A$ generates x' if $A \rhd x$.
- We write $S \rhd^* x$ if $A \rhd x$ for some $A \subseteq S$.
- $\langle S \rangle$ is the closure of $S \subseteq X$ i.e. $S \subseteq \langle S \rangle$ and $\langle S \rangle \rhd^* x$ implies $x \in \langle S \rangle$.

Example

Let X be a commutative ring and $\{a_1, \ldots, a_k\} \rhd x$ iff $a_1 \cdot x_1 + \ldots + a_k \cdot x_k = x$ for $x_1, \ldots, x_k \in X$. Then $I = \langle I \rangle$ iff I is an ideal of X.

We also consider the notion of an *open predicate* Q on subsets of X.

- Let $Q(x)$ be an arbitrary predicate on elements of X.
- Extend Q to subsets $S \subseteq X$ by defining $Q(S) := (\forall x \in S)Q(x)$.
- Note that $Q(S)$ and $T \subseteq S$ implies $Q(T)$.

Example

Let X be a commutative ring and define $Q(x)$ iff $x \neq 1$. Then $Q(S)$ iff $1 \notin S$.

Some basic definitions

We start of with an abstract generating relation $\rhd$:

- X is a set.
- $\rhd$ is a relation on $\mathcal{P}_{fin}(X) \times X$, where we say '$A$ generates x' if $A \rhd x$.
- We write $S \rhd^* x$ if $A \rhd x$ for some $A \subseteq S$.
- $\langle S \rangle$ is the closure of $S \subseteq X$ i.e. $S \subseteq \langle S \rangle$ and $\langle S \rangle \rhd^* x$ implies $x \in \langle S \rangle$.

Example

Let X be a commutative ring and $\{a_1, \ldots, a_k\} \rhd x$ iff $a_1 \cdot x_1 + \ldots + a_k \cdot x_k = x$ for $x_1, \ldots, x_k \in X$. Then $I = \langle I \rangle$ iff I is an ideal of X.

We also consider the notion of an *open predicate* Q on subsets of X.

- Let $\mathrm{Q}(x)$ be an arbitrary predicate on elements of X.
- Extend Q to subsets $S \subseteq X$ by defining $\mathrm{Q}(S) :\equiv (\forall x \in S)\mathrm{Q}(x)$.
- Note that $\mathrm{Q}(S)$ and $T \subseteq S$ implies $\mathrm{Q}(T)$.

Example

Let X be a commutative ring and define $\mathrm{Q}(x)$ iff $x \neq 1$. Then $\mathrm{Q}(S)$ iff $1 \notin S$.

Some basic definitions

We start of with an abstract generating relation $\rhd$:

- X is a set.
- $\rhd$ is a relation on $\mathcal{P}_{fin}(X) \times X$, where we say '$A$ generates x' if $A \rhd x$.
- We write $S \rhd^* x$ if $A \rhd x$ for some $A \subseteq S$.
- $\langle S \rangle$ is the closure of $S \subseteq X$ i.e. $S \subseteq \langle S \rangle$ and $\langle S \rangle \rhd^* x$ implies $x \in \langle S \rangle$.

Example

Let X be a commutative ring and $\{a_1, \ldots, a_k\} \rhd x$ iff $a_1 \cdot x_1 + \ldots + a_k \cdot x_k = x$ for $x_1, \ldots, x_k \in X$. Then $I = \langle I \rangle$ iff I is an ideal of X.

We also consider the notion of an *open predicate* Q on subsets of X.

- Let $Q(x)$ be an arbitrary predicate on elements of X.
- Extend Q to subsets $S \subseteq X$ by defining $Q(S) :\equiv (\forall x \in S) Q(x)$.
- Note that $Q(S)$ and $T \subseteq S$ implies $Q(T)$.

Example

Let X be a commutative ring and define $Q(x)$ iff $x \neq 1$. Then $Q(S)$ iff $1 \notin S$.

Some basic definitions

We start of with an abstract generating relation $\rhd$:

- X is a set.
- $\rhd$ is a relation on $\mathcal{P}_{fin}(X) \times X$, where we say '$A$ generates x' if $A \rhd x$.
- We write $S \rhd^* x$ if $A \rhd x$ for some $A \subseteq S$.
- $\langle S \rangle$ is the closure of $S \subseteq X$ i.e. $S \subseteq \langle S \rangle$ and $\langle S \rangle \rhd^* x$ implies $x \in \langle S \rangle$.

Example

Let X be a commutative ring and $\{a_1, \ldots, a_k\} \rhd x$ iff $a_1 \cdot x_1 + \ldots + a_k \cdot x_k = x$ for $x_1, \ldots, x_k \in X$. Then $I = \langle I \rangle$ iff I is an ideal of X.

We also consider the notion of an *open predicate* Q on subsets of X.

- Let $\mathrm{Q}(x)$ be an arbitrary predicate on elements of X.
- Extend Q to subsets $S \subseteq X$ by defining $\mathrm{Q}(S) := (\forall x \in S)\mathrm{Q}(x)$.
- Note that $\mathrm{Q}(S)$ and $T \subseteq S$ implies $\mathrm{Q}(T)$.

Example

Let X be a commutative ring and define $\mathrm{Q}(x)$ iff $x \neq 1$. Then $\mathrm{Q}(S)$ iff $1 \notin S$.

Some basic definitions

We start of with an abstract generating relation $\rhd$:

- X is a set.
- $\rhd$ is a relation on $\mathcal{P}_{fin}(X) \times X$, where we say '$A$ generates x' if $A \rhd x$.
- We write $S \rhd^* x$ if $A \rhd x$ for some $A \subseteq S$.
- $\langle S \rangle$ is the closure of $S \subseteq X$ i.e. $S \subseteq \langle S \rangle$ and $\langle S \rangle \rhd^* x$ implies $x \in \langle S \rangle$.

Example

Let X be a commutative ring and $\{a_1, \ldots, a_k\} \rhd x$ iff $a_1 \cdot x_1 + \ldots + a_k \cdot x_k = x$ for $x_1, \ldots, x_k \in X$. Then $I = \langle I \rangle$ iff I is an ideal of X.

We also consider the notion of an *open predicate* Q on subsets of X.

- Let $Q(x)$ be an arbitrary predicate on elements of X.
- Extend Q to subsets $S \subseteq X$ by defining $Q(S) := (\forall x \in S) Q(x)$.
- Note that $Q(S)$ and $T \subseteq S$ implies $Q(T)$.

Example

Let X be a commutative ring and define $Q(x)$ iff $x \neq 1$. Then $Q(S)$ iff $1 \notin S$.

Some basic definitions

We start of with an abstract generating relation $\triangleright$:

- X is a set.
- $\triangleright$ is a relation on $\mathcal{P}_{fin}(X) \times X$, where we say '$A$ generates x' if $A \triangleright x$.
- We write $S \triangleright^* x$ if $A \triangleright x$ for some $A \subseteq S$.
- $\langle S \rangle$ is the closure of $S \subseteq X$ i.e. $S \subseteq \langle S \rangle$ and $\langle S \rangle \triangleright^* x$ implies $x \in \langle S \rangle$.

Example

Let X be a commutative ring and $\{a_1, \ldots, a_k\} \triangleright x$ iff $a_1 \cdot x_1 + \ldots + a_k \cdot x_k = x$ for $x_1, \ldots, x_k \in X$. Then $I = \langle I \rangle$ iff I is an ideal of X.

We also consider the notion of an *open predicate* Q on subsets of X.

- Let $Q(x)$ be an arbitrary predicate on elements of X.
- Extend Q to subsets $S \subseteq X$ by defining $Q(S) := (\forall x \in S) Q(x)$.
- Note that $Q(S)$ and $T \subseteq S$ implies $Q(T)$.

Example

Let X be a commutative ring and define $Q(x)$ iff $x \neq 1$. Then $Q(S)$ iff $1 \notin S$.

Some basic definitions

We start of with an abstract generating relation $\rhd$:

- X is a set.
- $\rhd$ is a relation on $\mathcal{P}_{fin}(X) \times X$, where we say '$A$ generates x' if $A \rhd x$.
- We write $S \rhd^* x$ if $A \rhd x$ for some $A \subseteq S$.
- $\langle S \rangle$ is the closure of $S \subseteq X$ i.e. $S \subseteq \langle S \rangle$ and $\langle S \rangle \rhd^* x$ implies $x \in \langle S \rangle$.

Example

Let X be a commutative ring and $\{a_1, \ldots, a_k\} \rhd x$ iff $a_1 \cdot x_1 + \ldots + a_k \cdot x_k = x$ for $x_1, \ldots, x_k \in X$. Then $I = \langle I \rangle$ iff I is an ideal of X.

We also consider the notion of an *open predicate* Q on subsets of X.

- Let $Q(x)$ be an arbitrary predicate on elements of X.
- Extend Q to subsets $S \subseteq X$ by defining $Q(S) := (\forall x \in S) Q(x)$.
- Note that $Q(S)$ and $T \subseteq S$ implies $Q(T)$.

Example

Let X be a commutative ring and define $Q(x)$ iff $x \neq 1$. Then $Q(S)$ iff $1 \notin S$.

Some basic definitions

We start of with an abstract generating relation $\triangleright$:

- X is a set.
- $\triangleright$ is a relation on $\mathcal{P}_{fin}(X) \times X$, where we say '$A$ generates x' if $A \triangleright x$.
- We write $S \triangleright^* x$ if $A \triangleright x$ for some $A \subseteq S$.
- $\langle S \rangle$ is the closure of $S \subseteq X$ i.e. $S \subseteq \langle S \rangle$ and $\langle S \rangle \triangleright^* x$ implies $x \in \langle S \rangle$.

Example

Let X be a commutative ring and $\{a_1, \ldots, a_k\} \triangleright x$ iff $a_1 \cdot x_1 + \ldots + a_k \cdot x_k = x$ for $x_1, \ldots, x_k \in X$. Then $I = \langle I \rangle$ iff I is an ideal of X.

We also consider the notion of an *open predicate* Q on subsets of X.

- Let $\mathrm{Q}(x)$ be an arbitrary predicate on elements of X.
- Extend Q to subsets $S \subseteq X$ by defining $\mathrm{Q}(S) := (\forall x \in S)\mathrm{Q}(x)$.
- Note that $\mathrm{Q}(S)$ and $T \subseteq S$ implies $\mathrm{Q}(T)$.

Example

Let X be a commutative ring and define $\mathrm{Q}(x)$ iff $x \neq 1$. Then $\mathrm{Q}(S)$ iff $1 \notin S$.

Some basic definitions

We start of with an abstract generating relation $\triangleright$:

- X is a set.
- $\triangleright$ is a relation on $\mathcal{P}_{fin}(X) \times X$, where we say '$A$ generates x' if $A \triangleright x$.
- We write $S \triangleright^* x$ if $A \triangleright x$ for some $A \subseteq S$.
- $\langle S \rangle$ is the closure of $S \subseteq X$ i.e. $S \subseteq \langle S \rangle$ and $\langle S \rangle \triangleright^* x$ implies $x \in \langle S \rangle$.

Example

Let X be a commutative ring and $\{a_1, \ldots, a_k\} \triangleright x$ iff $a_1 \cdot x_1 + \ldots + a_k \cdot x_k = x$ for $x_1, \ldots, x_k \in X$. Then $I = \langle I \rangle$ iff I is an ideal of X.

We also consider the notion of an *open predicate* Q on subsets of X.

- Let $\mathrm{Q}(x)$ be an arbitrary predicate on elements of X.
- Extend Q to subsets $S \subseteq X$ by defining $\mathrm{Q}(S) := (\forall x \in S)\mathrm{Q}(x)$.
- Note that $\mathrm{Q}(S)$ and $T \subseteq S$ implies $\mathrm{Q}(T)$.

Example

Let X be a commutative ring and define $\mathrm{Q}(x)$ iff $x \neq 1$. Then $\mathrm{Q}(S)$ iff $1 \notin S$.

Some basic definitions

We start of with an abstract generating relation $\rhd$:

- X is a set.
- $\rhd$ is a relation on $\mathcal{P}_{fin}(X) \times X$, where we say '$A$ generates x' if $A \rhd x$.
- We write $S \rhd^* x$ if $A \rhd x$ for some $A \subseteq S$.
- $\langle S \rangle$ is the closure of $S \subseteq X$ i.e. $S \subseteq \langle S \rangle$ and $\langle S \rangle \rhd^* x$ implies $x \in \langle S \rangle$.

Example

Let X be a commutative ring and $\{a_1, \ldots, a_k\} \rhd x$ iff $a_1 \cdot x_1 + \ldots + a_k \cdot x_k = x$ for $x_1, \ldots, x_k \in X$. Then $I = \langle I \rangle$ iff I is an ideal of X.

We also consider the notion of an *open predicate* Q on subsets of X.

- Let $\mathrm{Q}(x)$ be an arbitrary predicate on elements of X.
- Extend Q to subsets $S \subseteq X$ by defining $\mathrm{Q}(S) := (\forall x \in S)\mathrm{Q}(x)$.
- Note that $\mathrm{Q}(S)$ and $T \subseteq S$ implies $\mathrm{Q}(T)$.

Example

Let X be a commutative ring and define $\mathrm{Q}(x)$ iff $x \neq 1$. Then $\mathrm{Q}(S)$ iff $1 \notin S$.

Some basic definitions

We start of with an abstract generating relation $\rhd$:

- X is a set.
- $\rhd$ is a relation on $\mathcal{P}_{fin}(X) \times X$, where we say '$A$ generates x' if $A \rhd x$.
- We write $S \rhd^* x$ if $A \rhd x$ for some $A \subseteq S$.
- $\langle S \rangle$ is the closure of $S \subseteq X$ i.e. $S \subseteq \langle S \rangle$ and $\langle S \rangle \rhd^* x$ implies $x \in \langle S \rangle$.

Example

Let X be a commutative ring and $\{a_1, \ldots, a_k\} \rhd x$ iff $a_1 \cdot x_1 + \ldots + a_k \cdot x_k = x$ for $x_1, \ldots, x_k \in X$. Then $I = \langle I \rangle$ iff I is an ideal of X.

We also consider the notion of an *open predicate* Q on subsets of X.

- Let $\mathrm{Q}(x)$ be an arbitrary predicate on elements of X.
- Extend Q to subsets $S \subseteq X$ by defining $\mathrm{Q}(S) := (\forall x \in S)\mathrm{Q}(x)$.
- Note that $\mathrm{Q}(S)$ and $T \subseteq S$ implies $\mathrm{Q}(T)$.

Example

Let X be a commutative ring and define $\mathrm{Q}(x)$ iff $x \neq 1$. Then $\mathrm{Q}(S)$ iff $1 \notin S$.

Some basic definitions

We start of with an abstract generating relation $\rhd$:

- X is a set.
- $\rhd$ is a relation on $\mathcal{P}_{fin}(X) \times X$, where we say '$A$ generates x' if $A \rhd x$.
- We write $S \rhd^* x$ if $A \rhd x$ for some $A \subseteq S$.
- $\langle S \rangle$ is the closure of $S \subseteq X$ i.e. $S \subseteq \langle S \rangle$ and $\langle S \rangle \rhd^* x$ implies $x \in \langle S \rangle$.

Example

Let X be a commutative ring and $\{a_1, \ldots, a_k\} \rhd x$ iff $a_1 \cdot x_1 + \ldots + a_k \cdot x_k = x$ for $x_1, \ldots, x_k \in X$. Then $I = \langle I \rangle$ iff I is an ideal of X.

We also consider the notion of an *open predicate* Q on subsets of X.

- Let $\mathrm{Q}(x)$ be an arbitrary predicate on elements of X.
- Extend Q to subsets $S \subseteq X$ by defining $\mathrm{Q}(S) := (\forall x \in S)\mathrm{Q}(x)$.
- Note that $\mathrm{Q}(S)$ and $T \subseteq S$ implies $\mathrm{Q}(T)$.

Example

Let X be a commutative ring and define $\mathrm{Q}(x)$ iff $x \neq 1$. Then $\mathrm{Q}(S)$ iff $1 \notin S$.

A general maximality principle

Theorem (P., Schuster & Wiesnet, WoLLIC '19)

Suppose that $Q(\langle\emptyset\rangle)$*. Then there exists some* $M \subseteq X$ *such that*

- M *is closed i.e.* $M = \langle M \rangle$,
- $Q(M)$ *holds,*
- $\neg Q(M \oplus x)$ *for all* $x \notin M$*, where* $M \oplus x := \langle M \cup \{x\} \rangle$.

We say that M *is maximal w.r.t.* $\rhd$ *and* Q.

Proof (sketch).

Define $\mathcal{U} := \{S \subseteq X \mid S \text{ is closed and } Q(S)\}$. Then $\langle\emptyset\rangle \in \mathcal{U}$ and $\mathcal{U}$ is chain complete w.r.t. $\subseteq$, therefore by Zorn's lemma it has a maximal element M.

M is closed and $Q(M)$ holds since $M \in \mathcal{U}$. For $x \notin M$ we have $M \subset M \oplus x$ and thus $M \oplus x \notin \mathcal{U}$. But since $M \oplus x$ is closed then we must have $\neg Q(M \oplus x)$. □

Example

Let X be a commutative ring with $0 \neq 1$. Continuing our previous example, we would have $S \in \mathcal{U}$ precisely when S is a *proper* ideal. Thus M is a maximal ideal.

A general maximality principle

Theorem (P., Schuster & Wiesnet, WoLLIC '19)

Suppose that $\mathsf{Q}(\langle\emptyset\rangle)$. *Then there exists some* $M \subseteq X$ *such that*

- M *is closed i.e.* $M = \langle M\rangle$,
- $\mathsf{Q}(M)$ *holds,*
- $\neg\mathsf{Q}(M \oplus x)$ *for all* $x \notin M$, *where* $M \oplus x := \langle M \cup \{x\}\rangle$.

We say that M *is maximal w.r.t.* $\rhd$ *and* Q.

Proof (sketch).

Define $\mathcal{U} := \{S \subseteq X \mid S \text{ is closed and } \mathsf{Q}(S)\}$. Then $\langle\emptyset\rangle \in \mathcal{U}$ and $\mathcal{U}$ is chain complete w.r.t. $\subseteq$, therefore by Zorn's lemma it has a maximal element M.

M is closed and $\mathsf{Q}(M)$ holds since $M \in \mathcal{U}$. For $x \notin M$ we have $M \subset M \oplus x$ and thus $M \oplus x \notin \mathcal{U}$. But since $M \oplus x$ is closed then we must have $\neg\mathsf{Q}(M \oplus x)$. □

Example

Let X be a commutative ring with $0 \neq 1$. Continuing our previous example, we would have $S \in \mathcal{U}$ precisely when S is a *proper* ideal. Thus M is a maximal ideal.

A general maximality principle

Theorem (P., Schuster & Wiesnet, WoLLIC '19)

Suppose that $Q(\langle\emptyset\rangle)$. *Then there exists some* $M \subseteq X$ *such that*

- M *is closed i.e.* $M = \langle M\rangle$,
- $Q(M)$ *holds,*
- $\neg Q(M \oplus x)$ *for all* $x \notin M$, *where* $M \oplus x := \langle M \cup \{x\}\rangle$.

We say that M *is maximal w.r.t.* $\rhd$ *and* Q.

Proof (sketch).

Define $\mathcal{U} := \{S \subseteq X \mid S \text{ is closed and } Q(S)\}$. Then $\langle\emptyset\rangle \in \mathcal{U}$ and $\mathcal{U}$ is chain complete w.r.t. $\subseteq$, therefore by Zorn's lemma it has a maximal element M.

M is closed and $Q(M)$ holds since $M \in \mathcal{U}$. For $x \notin M$ we have $M \subset M \oplus x$ and thus $M \oplus x \notin \mathcal{U}$. But since $M \oplus x$ is closed then we must have $\neg Q(M \oplus x)$. □

Example

Let X be a commutative ring with $0 \neq 1$. Continuing our previous example, we would have $S \in \mathcal{U}$ precisely when S is a *proper* ideal. Thus M is a maximal ideal.

A general maximality principle

Theorem (P., Schuster & Wiesnet, WoLLIC '19)

Suppose that $\mathrm{Q}(\langle\emptyset\rangle)$. *Then there exists some* $M \subseteq X$ *such that*

- M *is closed i.e.* $M = \langle M\rangle$,
- $\mathrm{Q}(M)$ *holds,*
- $\neg\mathrm{Q}(M \oplus x)$ *for all* $x \notin M$, *where* $M \oplus x := \langle M \cup \{x\}\rangle$.

We say that M *is maximal w.r.t.* $\rhd$ *and* Q.

Proof (sketch).

Define $\mathcal{U} := \{S \subseteq X \mid S \text{ is closed and } \mathrm{Q}(S)\}$. Then $\langle\emptyset\rangle \in \mathcal{U}$ and $\mathcal{U}$ is chain complete w.r.t. $\subseteq$, therefore by Zorn's lemma it has a maximal element M.

M is closed and $\mathrm{Q}(M)$ holds since $M \in \mathcal{U}$. For $x \notin M$ we have $M \subset M \oplus x$ and thus $M \oplus x \notin \mathcal{U}$. But since $M \oplus x$ is closed then we must have $\neg\mathrm{Q}(M \oplus x)$. □

Example

Let X be a commutative ring with $0 \neq 1$. Continuing our previous example, we would have $S \in \mathcal{U}$ precisely when S is a *proper* ideal. Thus M is a maximal ideal.

A general maximality principle

Theorem (P., Schuster & Wiesnet, WoLLIC '19)

Suppose that $\mathsf{Q}(\langle\emptyset\rangle)$*. Then there exists some* $M \subseteq X$ *such that*

- M *is closed i.e.* $M = \langle M\rangle$*,*
- $\mathsf{Q}(M)$ *holds,*
- $\neg\mathsf{Q}(M \oplus x)$ *for all* $x \notin M$*, where* $M \oplus x := \langle M \cup \{x\}\rangle$*.*

We say that M *is maximal w.r.t.* $\rhd$ *and* Q*.*

Proof (sketch).

Define $\mathcal{U} := \{S \subseteq X \mid S \text{ is closed and } \mathsf{Q}(S)\}$. Then $\langle\emptyset\rangle \in \mathcal{U}$ and $\mathcal{U}$ is chain complete w.r.t. $\subseteq$, therefore by Zorn's lemma it has a maximal element M.

M is closed and $\mathsf{Q}(M)$ holds since $M \in \mathcal{U}$. For $x \notin M$ we have $M \subset M \oplus x$ and thus $M \oplus x \notin \mathcal{U}$. But since $M \oplus x$ is closed then we must have $\neg\mathsf{Q}(M \oplus x)$. □

Example

Let X be a commutative ring with $0 \neq 1$. Continuing our previous example, we would have $S \in \mathcal{U}$ precisely when S is a *proper* ideal. Thus M is a maximal ideal.

A logical analysis of the countable case

From now on, suppose that $X := \{x_n \mid n \in \mathbb{N}\}$ is countable.
Define $[S](n) := S \cap \{x_m \mid m < n\}$.

Theorem

Suppose that $M \subseteq X$ satisfies

$$x_n \in M \Leftrightarrow Q([M](n) \oplus x_n)$$

for all $n \in \mathbb{N}$. Then $Q(\langle \emptyset \rangle)$ implies that M is maximal.

Idea. In the countable case, maximal objects can be constructed in a sequential fashion (formally, using dependent choice).

Example

Let X be a commutative ring with $0 \neq 1$ and suppose that M satisfies

$$x_n \in M \Leftrightarrow \forall b \in X^*, y \in X(b \cdot [M](n) + y \cdot x_n \neq 1).$$

Then M is a maximal ideal.

This is a standard trick in reverse math (cf. Lemma III.5.4. of Simpson's Reverse Maths book, where a similar argument is used to show that the existence of maximal ideals in countable rings is provable in ACA$_0$).

A logical analysis of the countable case

From now on, suppose that $X := \{x_n \mid n \in \mathbb{N}\}$ is countable.
Define $[S](n) := S \cap \{x_m \mid m < n\}$.

Theorem

Suppose that $M \subseteq X$ satisfies

$$x_n \in M \Leftrightarrow Q([M](n) \oplus x_n)$$

for all $n \in \mathbb{N}$. Then $Q(\langle \emptyset \rangle)$ implies that M is maximal.

Idea. In the countable case, maximal objects can be constructed in a sequential fashion (formally, using dependent choice).

Example

Let X be a commutative ring with $0 \neq 1$ and suppose that M satisfies

$$x_n \in M \Leftrightarrow \forall b \in X^*, y \in X(b \cdot [M](n) + y \cdot x_n \neq 1).$$

Then M is a maximal ideal.

This is a standard trick in reverse math (cf. Lemma III.5.4. of Simpson's Reverse Maths book, where a similar argument is used to show that the existence of maximal ideals in countable rings is provable in ACA_0).

A logical analysis of the countable case

From now on, suppose that $X := \{x_n \mid n \in \mathbb{N}\}$ is countable.
Define $[S](n) := S \cap \{x_m \mid m < n\}$.

Theorem

Suppose that $M \subseteq X$ satisfies

$$x_n \in M \Leftrightarrow \mathrm{Q}([M](n) \oplus x_n)$$

for all $n \in \mathbb{N}$. Then $\mathrm{Q}(\langle\emptyset\rangle)$ implies that M is maximal.

Idea. In the countable case, maximal objects can be constructed in a sequential fashion (formally, using dependent choice).

Example

Let X be a commutative ring with $0 \neq 1$ and suppose that M satisfies

$$x_n \in M \Leftrightarrow \forall b \in X^*, y \in X(b \cdot [M](n) + y \cdot x_n \neq 1).$$

Then M is a maximal ideal.

This is a standard trick in reverse math (cf. Lemma III.5.4. of Simpson's Reverse Maths book, where a similar argument is used to show that the existence of maximal ideals in countable rings is provable in ACA_0).

A logical analysis of the countable case

From now on, suppose that $X := \{x_n \mid n \in \mathbb{N}\}$ is countable.
Define $[S](n) := S \cap \{x_m \mid m < n\}$.

Theorem

Suppose that $M \subseteq X$ satisfies

$$x_n \in M \Leftrightarrow \mathrm{Q}([M](n) \oplus x_n)$$

for all $n \in \mathbb{N}$. Then $\mathrm{Q}(\langle\emptyset\rangle)$ implies that M is maximal.

Idea. In the countable case, maximal objects can be constructed in a sequential fashion (formally, using dependent choice).

Example

Let X be a commutative ring with $0 \neq 1$ and suppose that M satisfies

$$x_n \in M \Leftrightarrow \forall b \in X^*, y \in X(b \cdot [M](n) + y \cdot x_n \neq 1).$$

Then M is a maximal ideal.

This is a standard trick in reverse math (cf. Lemma III.5.4. of Simpson's Reverse Maths book, where a similar argument is used to show that the existence of maximal ideals in countable rings is provable in ACA_0).

A logical analysis of the countable case

From now on, suppose that $X := \{x_n \mid n \in \mathbb{N}\}$ is countable.
Define $[S](n) := S \cap \{x_m \mid m < n\}$.

Theorem

Suppose that $M \subseteq X$ satisfies

$$x_n \in M \Leftrightarrow \mathrm{Q}([M](n) \oplus x_n)$$

for all $n \in \mathbb{N}$. Then $\mathrm{Q}(\langle\emptyset\rangle)$ implies that M is maximal.

Idea. In the countable case, maximal objects can be constructed in a sequential fashion (formally, using dependent choice).

Example

Let X be a commutative ring with $0 \neq 1$ and suppose that M satisfies

$$x_n \in M \Leftrightarrow \forall b \in X^*, y \in X(b \cdot [M](n) + y \cdot x_n \neq 1).$$

Then M is a maximal ideal.

This is a standard trick in reverse math (cf. Lemma III.5.4. of Simpson's Reverse Maths book, where a similar argument is used to show that the existence of maximal ideals in countable rings is provable in ACA_0).

A logical analysis of the countable case

From now on, suppose that $X := \{x_n \mid n \in \mathbb{N}\}$ is countable.
Define $[S](n) := S \cap \{x_m \mid m < n\}$.

Theorem

Suppose that $M \subseteq X$ satisfies

$$x_n \in M \Leftrightarrow \mathrm{Q}([M](n) \oplus x_n)$$

for all $n \in \mathbb{N}$. Then $\mathrm{Q}(\langle \emptyset \rangle)$ implies that M is maximal.

Idea. In the countable case, maximal objects can be constructed in a sequential fashion (formally, using dependent choice).

Example

Let X be a commutative ring with $0 \neq 1$ and suppose that M satisfies

$$x_n \in M \Leftrightarrow \forall b \in X^*, y \in X(b \cdot [M](n) + y \cdot x_n \neq 1).$$

Then M is a maximal ideal.

This is a standard trick in reverse math (cf. Lemma III.5.4. of Simpson's Reverse Maths book, where a similar argument is used to show that the existence of maximal ideals in countable rings is provable in ACA_0).

The logical complexity of $Q(\langle S \rangle)$

Suppose that $Q(x)$ is a Π^0_1-formula, and $A \rhd x$ can be encoded as a Σ^0_1-formula. Then $Q(\langle S \rangle)$ can be encoded as a Π^0_1-formula.

- $x \in \langle S \rangle$ iff there exists some finite derivation tree t whose leaves are elements of S and whose nodes represent instances of $\rhd$.
- if $A \rhd x$ is Σ^0_1, then being a derivation tree is also Σ^0_1, and hence so is the existence of a derivation tree i.e. $x \in \langle S \rangle$.
- $Q(\langle S \rangle)$ is Π^0_1 since

$$Q(\langle S \rangle) \Leftrightarrow (\forall x)(\underbrace{x \in \langle S \rangle}_{\Sigma^0_1} \Rightarrow \underbrace{Q(x)}_{\Pi^0_1})$$

Theorem

The existence of a maximal structure can be encoded

$$(\exists M)(\forall n)\left(x_n \in M \Leftrightarrow (\forall p) R_{[M](n) \cup \{x_n\}}(p)\right)$$

for some suitable decidable predicate $R_A(p)$.

The logical complexity of $Q(\langle S \rangle)$

Suppose that $Q(x)$ is a Π^0_1-formula, and $A \rhd x$ can be encoded as a Σ^0_1-formula. Then $Q(\langle S \rangle)$ can be encoded as a Π^0_1-formula.

- $x \in \langle S \rangle$ iff there exists some finite derivation tree t whose leaves are elements of S and whose nodes represent instances of $\rhd$.
- if $A \rhd x$ is Σ^0_1, then being a derivation tree is also Σ^0_1, and hence so is the existence of a derivation tree i.e. $x \in \langle S \rangle$.
- $Q(\langle S \rangle)$ is Π^0_1 since

$$Q(\langle S \rangle) \Leftrightarrow (\forall x)(\underbrace{x \in \langle S \rangle}_{\Sigma^0_1} \Rightarrow \underbrace{Q(x)}_{\Pi^0_1})$$

Theorem

The existence of a maximal structure can be encoded

$$(\exists M)(\forall n)\left(x_n \in M \Leftrightarrow (\forall p) R_{[M](n) \cup \{x_n\}}(p)\right)$$

for some suitable decidable predicate $R_A(p)$.

The logical complexity of $Q(\langle S\rangle)$

Suppose that $Q(x)$ is a Π^0_1-formula, and $A \rhd x$ can be encoded as a Σ^0_1-formula. Then $Q(\langle S\rangle)$ can be encoded as a Π^0_1-formula.

- $x \in \langle S\rangle$ iff there exists some finite derivation tree t whose leaves are elements of S and whose nodes represent instances of $\rhd$.
- if $A \rhd x$ is Σ^0_1, then being a derivation tree is also Σ^0_1, and hence so is the existence of a derivation tree i.e. $x \in \langle S\rangle$.
- $Q(\langle S\rangle)$ is Π^0_1 since

$$Q(\langle S\rangle) \Leftrightarrow (\forall x)(\underbrace{x \in \langle S\rangle}_{\Sigma^0_1} \Rightarrow \underbrace{Q(x)}_{\Pi^0_1})$$

Theorem

The existence of a maximal structure can be encoded

$$(\exists M)(\forall n)\left(x_n \in M \Leftrightarrow (\forall p)R_{[M](n)\cup\{x_n\}}(p)\right)$$

for some suitable decidable predicate $R_A(p)$.

The logical complexity of $Q(\langle S \rangle)$

Suppose that $Q(x)$ is a Π^0_1-formula, and $A \rhd x$ can be encoded as a Σ^0_1-formula. Then $Q(\langle S \rangle)$ can be encoded as a Π^0_1-formula.

- $x \in \langle S \rangle$ iff there exists some finite derivation tree t whose leaves are elements of S and whose nodes represent instances of $\rhd$.
- if $A \rhd x$ is Σ^0_1, then being a derivation tree is also Σ^0_1, and hence so is the existence of a derivation tree i.e. $x \in \langle S \rangle$.
- $Q(\langle S \rangle)$ is Π^0_1 since

$$Q(\langle S \rangle) \Leftrightarrow (\forall x)(\underbrace{x \in \langle S \rangle}_{\Sigma^0_1} \Rightarrow \underbrace{Q(x)}_{\Pi^0_1})$$

Theorem

The existence of a maximal structure can be encoded

$$(\exists M)(\forall n)\left(x_n \in M \Leftrightarrow (\forall p) R_{[M](n) \cup \{x_n\}}(p)\right)$$

for some suitable decidable predicate $R_A(p)$.

The logical complexity of $Q(\langle S \rangle)$

Suppose that $Q(x)$ is a Π^0_1-formula, and $A \rhd x$ can be encoded as a Σ^0_1-formula. Then $Q(\langle S \rangle)$ can be encoded as a Π^0_1-formula.

- $x \in \langle S \rangle$ iff there exists some finite derivation tree t whose leaves are elements of S and whose nodes represent instances of $\rhd$.
- if $A \rhd x$ is Σ^0_1, then being a derivation tree is also Σ^0_1, and hence so is the existence of a derivation tree i.e. $x \in \langle S \rangle$.
- $Q(\langle S \rangle)$ is Π^0_1 since

$$Q(\langle S \rangle) \Leftrightarrow (\forall x)(\underbrace{x \in \langle S \rangle}_{\Sigma^0_1} \Rightarrow \underbrace{Q(x)}_{\Pi^0_1})$$

Theorem

The existence of a maximal structure can be encoded

$$(\exists M)(\forall n)\left(x_n \in M \Leftrightarrow (\forall p) R_{[M](n) \cup \{x_n\}}(p)\right)$$

for some suitable decidable predicate $R_A(p)$.

The logical complexity of $Q(\langle S\rangle)$

Suppose that $Q(x)$ is a Π^0_1-formula, and $A \rhd x$ can be encoded as a Σ^0_1-formula. Then $Q(\langle S\rangle)$ can be encoded as a Π^0_1-formula.

- $x \in \langle S\rangle$ iff there exists some finite derivation tree t whose leaves are elements of S and whose nodes represent instances of $\rhd$.
- if $A \rhd x$ is Σ^0_1, then being a derivation tree is also Σ^0_1, and hence so is the existence of a derivation tree i.e. $x \in \langle S\rangle$.
- $Q(\langle S\rangle)$ is Π^0_1 since

$$Q(\langle S\rangle) \Leftrightarrow (\forall x)(\underbrace{x \in \langle S\rangle}_{\Sigma^0_1} \Rightarrow \underbrace{Q(x)}_{\Pi^0_1})$$

Theorem

The existence of a maximal structure can be encoded

$$(\exists M)(\forall n)\left(x_n \in M \Leftrightarrow (\forall p)R_{[M](n)\cup\{x_n\}}(p)\right)$$

for some suitable decidable predicate $R_A(p)$.

Applying the classical functional interpretation

$$(\exists M)(\forall n)\left(x_n \in M \Leftrightarrow (\forall p)R_{[M](n)\cup\{x_n\}}(p)\right)$$

Written out fully we get:

$$(\exists M)(\forall n)\begin{pmatrix} x_n \in M \Rightarrow (\forall p)R_{[M](n)\cup\{x_n\}}(p) \\ \wedge\ x_n \notin M \Rightarrow (\exists p)\neg R_{[M](n)\cup\{x_n\}}(p) \end{pmatrix}$$

Bringing the quantifiers to the front:

$$(\exists M,f)(\forall n,p)\begin{pmatrix} x_n \in M \Rightarrow R_{[M](n)\cup\{x_n\}}(p) \\ \wedge\ x_n \notin M \Rightarrow \neg R_{[M](n)\cup\{x_n\}}(f(n)) \end{pmatrix}$$

The (partial) functional interpretation is then:

$$(\forall \omega,\phi)(\exists M,f)\begin{pmatrix} x_{\omega(M,f)} \in M \Rightarrow R_{[M](\omega(M,f))\cup\{x_{\omega(M,f)}\}}(\phi(M,f)) \\ \wedge\ x_{\omega(M,f)} \notin M \Rightarrow \neg R_{[M](\omega(M,f))\cup\{x_{\omega(M,f)}\}}(f(\omega(M,f))) \end{pmatrix}$$

Applying the classical functional interpretation

$$(\exists M)(\forall n)\left(x_n \in M \Leftrightarrow (\forall p)R_{[M](n)\cup\{x_n\}}(p)\right)$$

Written out fully we get:

$$(\exists M)(\forall n)\begin{pmatrix} x_n \in M \Rightarrow (\forall p)R_{[M](n)\cup\{x_n\}}(p) \\ \wedge\ x_n \notin M \Rightarrow (\exists p)\neg R_{[M](n)\cup\{x_n\}}(p) \end{pmatrix}$$

Bringing the quantifiers to the front:

$$(\exists M, f)(\forall n, p)\begin{pmatrix} x_n \in M \Rightarrow R_{[M](n)\cup\{x_n\}}(p) \\ \wedge\ x_n \notin M \Rightarrow \neg R_{[M](n)\cup\{x_n\}}(f(n)) \end{pmatrix}$$

The (partial) functional interpretation is then:

$$(\forall \omega, \phi)(\exists M, f)\begin{pmatrix} x_{\omega(M,f)} \in M \Rightarrow R_{[M](\omega(M,f))\cup\{x_{\omega(M,f)}\}}(\phi(M,f)) \\ \wedge\ x_{\omega(M,f)} \notin M \Rightarrow \neg R_{[M](\omega(M,f))\cup\{x_{\omega(M,f)}\}}(f(\omega(M,f))) \end{pmatrix}$$

Applying the classical functional interpretation

$$(\exists M)(\forall n)\left(x_n \in M \Leftrightarrow (\forall p) R_{[M](n)\cup\{x_n\}}(p)\right)$$

Written out fully we get:

$$(\exists M)(\forall n)\left(\begin{array}{c} x_n \in M \Rightarrow (\forall p) R_{[M](n)\cup\{x_n\}}(p) \\ \wedge\ x_n \notin M \Rightarrow (\exists p) \neg R_{[M](n)\cup\{x_n\}}(p) \end{array}\right)$$

Bringing the quantifiers to the front:

$$(\exists M, f)(\forall n, p)\left(\begin{array}{c} x_n \in M \Rightarrow R_{[M](n)\cup\{x_n\}}(p) \\ \wedge\ x_n \notin M \Rightarrow \neg R_{[M](n)\cup\{x_n\}}(f(n)) \end{array}\right)$$

The (partial) functional interpretation is then:

$$(\forall \omega, \phi)(\exists M, f)\left(\begin{array}{c} x_{\omega(M,f)} \in M \Rightarrow R_{[M](\omega(M,f))\cup\{x_{\omega(M,f)}\}}(\phi(M,f)) \\ \wedge\ x_{\omega(M,f)} \notin M \Rightarrow \neg R_{[M](\omega(M,f))\cup\{x_{\omega(M,f)}\}}(f(\omega(M,f))) \end{array}\right)$$

Applying the classical functional interpretation

$$(\exists M)(\forall n)\left(x_n \in M \Leftrightarrow (\forall p)R_{[M](n)\cup\{x_n\}}(p)\right)$$

Written out fully we get:

$$(\exists M)(\forall n)\begin{pmatrix} x_n \in M \Rightarrow (\forall p)R_{[M](n)\cup\{x_n\}}(p) \\ \wedge\ x_n \notin M \Rightarrow (\exists p)\neg R_{[M](n)\cup\{x_n\}}(p) \end{pmatrix}$$

Bringing the quantifiers to the front:

$$(\exists M, f)(\forall n, p)\begin{pmatrix} x_n \in M \Rightarrow R_{[M](n)\cup\{x_n\}}(p) \\ \wedge\ x_n \notin M \Rightarrow \neg R_{[M](n)\cup\{x_n\}}(f(n)) \end{pmatrix}$$

The (partial) functional interpretation is then:

$$(\forall \omega, \phi)(\exists M, f)\begin{pmatrix} x_{\omega(M,f)} \in M \Rightarrow R_{[M](\omega(M,f))\cup\{x_{\omega(M,f)}\}}(\phi(M,f)) \\ \wedge\ x_{\omega(M,f)} \notin M \Rightarrow \neg R_{[M](\omega(M,f))\cup\{x_{\omega(M,f)}\}}(f(\omega(M,f))) \end{pmatrix}$$

Applying the classical functional interpretation

$$(\exists M)(\forall n)\left(x_n \in M \Leftrightarrow (\forall p)R_{[M](n)\cup\{x_n\}}(p)\right)$$

Written out fully we get:

$$(\exists M)(\forall n)\left(\begin{array}{c} x_n \in M \Rightarrow (\forall p)R_{[M](n)\cup\{x_n\}}(p) \\ \wedge\; x_n \notin M \Rightarrow (\exists p)\neg R_{[M](n)\cup\{x_n\}}(p) \end{array}\right)$$

Bringing the quantifiers to the front:

$$(\exists M, f)(\forall n, p)\left(\begin{array}{c} x_n \in M \Rightarrow R_{[M](n)\cup\{x_n\}}(p) \\ \wedge\; x_n \notin M \Rightarrow \neg R_{[M](n)\cup\{x_n\}}(f(n)) \end{array}\right)$$

The (partial) functional interpretation is then:

$$(\forall \omega, \phi)(\exists M, f)\left(\begin{array}{c} x_{\omega(M,f)} \in M \Rightarrow R_{[M](\omega(M,f))\cup\{x_{\omega(M,f)}\}}(\phi(M,f)) \\ \wedge\; x_{\omega(M,f)} \notin M \Rightarrow \neg R_{[M](\omega(M,f))\cup\{x_{\omega(M,f)}\}}(f(\omega(M,f))) \end{array}\right)$$

Approximate maximal objects (corresponds to $\forall\phi\exists x P(x,\phi x)$)

$$(\forall\omega,\phi)(\exists M,f)\left(\begin{array}{c} x_{\omega(M,f)}\in M\Rightarrow R_{[M](\omega(M,f))\cup\{x_{\omega(M,f)}\}}(\phi(M,f)) \\ \wedge\; x_{\omega(M,f)}\notin M\Rightarrow \neg R_{[M](\omega(M,f))\cup\{x_{\omega(M,f)}\}}(f(\omega(M,f)))\end{array}\right)$$

The functional interpretation of maximality inspires the following definition:

Definition

Given functionals (ω,ϕ), we say that $M\subseteq X$ and $f:\mathrm{dom}(X\backslash M)\to\mathbb{N}$ constitute an *approximate maximal object* relative to (ω,ϕ) if they satisfy

- $x_n\in M\Rightarrow R_{[M](n)\cup\{x_n\}}(p)$
- $x_n\notin M\Rightarrow\neg R_{[M](n)\cup\{x_n\}}(f(n))$

for all $n\leq\omega(M,f)$ and $p=\phi(M,f)$.

We now design a sequential algorithm

$$M_0,f_0\mapsto M_1,f_1\mapsto\ldots\mapsto M_k,f_k$$

which computes approximate maximal objects for any (ω,ϕ).

Approximate maximal objects (corresponds to $\forall\phi\exists x P(x, \phi x)$)

$$(\forall\omega,\phi)(\exists M,f)\left(\begin{array}{c} x_{\omega(M,f)}\in M \Rightarrow R_{[M](\omega(M,f))\cup\{x_{\omega(M,f)}\}}(\phi(M,f)) \\ \wedge\ x_{\omega(M,f)}\notin M \Rightarrow \neg R_{[M](\omega(M,f))\cup\{x_{\omega(M,f)}\}}(f(\omega(M,f))) \end{array}\right)$$

The functional interpretation of maximality inspires the following definition:

Definition

Given functionals (ω, ϕ), we say that $M \subseteq X$ and $f : \mathrm{dom}(X\backslash M) \to \mathbb{N}$ constitute an *approximate maximal object* relative to (ω, ϕ) if they satisfy

- $x_n \in M \Rightarrow R_{[M](n)\cup\{x_n\}}(p)$
- $x_n \notin M \Rightarrow \neg R_{[M](n)\cup\{x_n\}}(f(n))$

for all $n \leq \omega(M, f)$ and $p = \phi(M, f)$.

We now design a sequential algorithm

$$M_0, f_0 \mapsto M_1, f_1 \mapsto \ldots \mapsto M_k, f_k$$

which computes approximate maximal objects for any (ω, ϕ).

Approximate maximal objects (corresponds to $\forall\phi\exists x P(x, \phi x)$)

$$(\forall\omega, \phi)(\exists M, f)\left(\begin{array}{c} x_{\omega(M,f)} \in M \Rightarrow R_{[M](\omega(M,f))\cup\{x_{\omega(M,f)}\}}(\phi(M,f)) \\ \wedge\, x_{\omega(M,f)} \notin M \Rightarrow \neg R_{[M](\omega(M,f))\cup\{x_{\omega(M,f)}\}}(f(\omega(M,f))) \end{array}\right)$$

The functional interpretation of maximality inspires the following definition:

Definition

Given functionals (ω, ϕ), we say that $M \subseteq X$ and $f : \mathrm{dom}(X \backslash M) \to \mathbb{N}$ constitute an *approximate maximal object* relative to (ω, ϕ) if they satisfy

- $x_n \in M \Rightarrow R_{[M](n)\cup\{x_n\}}(p)$
- $x_n \notin M \Rightarrow \neg R_{[M](n)\cup\{x_n\}}(f(n))$

for all $n \leq \omega(M, f)$ and $p = \phi(M, f)$.

We now design a sequential algorithm

$$M_0, f_0 \mapsto M_1, f_1 \mapsto \ldots \mapsto M_k, f_k$$

which computes approximate maximal objects for any (ω, ϕ).

Approximate maximal objects (corresponds to $\forall\phi\exists x P(x, \phi x)$)

$$(\forall \omega, \phi)(\exists M, f) \left(\begin{array}{l} \quad x_{\omega(M,f)} \in M \Rightarrow R_{[M](\omega(M,f))\cup\{x_{\omega(M,f)}\}}(\phi(M,f)) \\ \wedge\, x_{\omega(M,f)} \notin M \Rightarrow \neg R_{[M](\omega(M,f))\cup\{x_{\omega(M,f)}\}}(f(\omega(M,f))) \end{array} \right)$$

The functional interpretation of maximality inspires the following definition:

Definition

Given functionals (ω, ϕ), we say that $M \subseteq X$ and $f : \mathrm{dom}(X \backslash M) \to \mathbb{N}$ constitute an *approximate maximal object* relative to (ω, ϕ) if they satisfy

- $x_n \in M \Rightarrow R_{[M](n)\cup\{x_n\}}(p)$
- $x_n \notin M \Rightarrow \neg R_{[M](n)\cup\{x_n\}}(f(n))$

for all $n \leq \omega(M, f)$ and $p = \phi(M, f)$.

We now design a sequential algorithm

$$M_0, f_0 \mapsto M_1, f_1 \mapsto \ldots \mapsto M_k, f_k$$

which computes approximate maximal objects for any (ω, ϕ).

The algorithm (corresponds to finding some F such that $\forall\phi P(F, \phi F)$)

A state s is defined to be a function $\mathbb{N} \to \{(*)\} + \mathbb{N}$. Any state induces a set $M[s]$ and a function $f : \mathrm{dom}(X \backslash M[s]) \to \mathbb{N}$ via

$$M[s] := \{x_n \mid s_i(n) = (*)\}$$
$$f[s] := \lambda n \; . \; s_i(n)$$

Define the sequential algorithm $\{s_i\}_{i \in \mathbb{N}}$ by $s_0(n) = (*)$ (i.e. $M[s_0] = X$) and

1. set $n_i, p_i := \omega(M[s_i], f[s_i]), \phi(M[s_i], f[s_i])$
2. search from 0 up to n_i for some n such that
    - $x_n \in M[s_i]$
    - $\neg R_{[M[s_i]](n) \cup \{x_n\}}(p_i)$

    1. if none is found, terminate
    2. else, set $s_{i+1} := [s_i](n) :: p_i :: \lambda k \; . \; (*)$

Theorem

Whenever the above algorithm is run on continuous parameters (ω, ϕ), it terminates in some state s_j where $M[s_j], f[s_j]$ form an approximate maximal object w.r.t. (ω, ϕ).

The algorithm (corresponds to finding some F such that $\forall \phi P(F, \phi F)$)

A state s is defined to be a function $\mathbb{N} \to \{(*)\} + \mathbb{N}$. Any state induces a set $M[s]$ and a function $f : \mathrm{dom}(X \backslash M[s]) \to \mathbb{N}$ via

$$M[s] := \{x_n \mid s_i(n) = (*)\}$$
$$f[s] := \lambda n \;.\; s_i(n)$$

Define the sequential algorithm $\{s_i\}_{i \in \mathbb{N}}$ by $s_0(n) = (*)$ (i.e. $M[s_0] = X$) and

1. set $n_i, p_i := \omega(M[s_i], f[s_i]), \phi(M[s_i], f[s_i])$
2. search from 0 up to n_i for some n such that
   - $x_n \in M[s_i]$
   - $\neg R_{[M[s_i]](n) \cup \{x_n\}}(p_i)$

1. if none is found, terminate

2. else, set $s_{i+1} := [s_i](n) :: p_i :: \lambda k \;.\; (*)$

Theorem

Whenever the above algorithm is run on continuous parameters (ω, ϕ), it terminates in some state s_j where $M[s_j], f[s_j]$ form an approximate maximal object w.r.t. (ω, ϕ).

The algorithm (corresponds to finding some F such that $\forall \phi P(F, \phi F)$)

A state s is defined to be a function $\mathbb{N} \to \{(*)\} + \mathbb{N}$. Any state induces a set $M[s]$ and a function $f : \mathrm{dom}(X \backslash M[s]) \to \mathbb{N}$ via

$$\begin{aligned} M[s] &:= \{x_n \mid s_i(n) = (*)\} \\ f[s] &:= \lambda n \;.\; s_i(n) \end{aligned}$$

Define the sequential algorithm $\{s_i\}_{i \in \mathbb{N}}$ by $s_0(n) = (*)$ (i.e. $M[s_0] = X$) and

1. set $n_i, p_i := \omega(M[s_i], f[s_i]), \phi(M[s_i], f[s_i])$
2. search from 0 up to n_i for some n such that
    - $x_n \in M[s_i]$
    - $\neg R_{[M[s_i]](n) \cup \{x_n\}}(p_i)$

1. if none is found, terminate
2. else, set $s_{i+1} := [s_i](n) :: p_i :: \lambda k \;.\; (*)$

Theorem

Whenever the above algorithm is run on continuous parameters (ω, ϕ), it terminates in some state s_j where $M[s_j], f[s_j]$ form an approximate maximal object w.r.t. (ω, ϕ).

The algorithm (corresponds to finding some F such that $\forall \phi P(F, \phi F)$)

A state s is defined to be a function $\mathbb{N} \to \{(*)\} + \mathbb{N}$. Any state induces a set $M[s]$ and a function $f : \mathrm{dom}(X \backslash M[s]) \to \mathbb{N}$ via

$$\begin{aligned} M[s] &:= \{x_n \mid s_i(n) = (*)\} \\ f[s] &:= \lambda n \,.\, s_i(n) \end{aligned}$$

Define the sequential algorithm $\{s_i\}_{i \in \mathbb{N}}$ by $s_0(n) = (*)$ (i.e. $M[s_0] = X$) and

1. set $n_i, p_i := \omega(M[s_i], f[s_i]), \phi(M[s_i], f[s_i])$
2. search from 0 up to n_i for some n such that
    - $x_n \in M[s_i]$
    - $\neg R_{[M[s_i]](n) \cup \{x_n\}}(p_i)$

    1. if none is found, terminate
    2. else, set $s_{i+1} := [s_i](n) :: p_i :: \lambda k \,.\, (*)$

Theorem

Whenever the above algorithm is run on continuous parameters (ω, ϕ), it terminates in some state s_j where $M[s_j], f[s_j]$ form an approximate maximal object w.r.t. (ω, ϕ).

The algorithm (corresponds to finding some F such that $\forall \phi P(F, \phi F)$)

A state s is defined to be a function $\mathbb{N} \to \{(*)\} + \mathbb{N}$. Any state induces a set $M[s]$ and a function $f : \mathrm{dom}(X \backslash M[s]) \to \mathbb{N}$ via

$$\begin{aligned} M[s] &:= \{x_n \mid s_i(n) = (*)\} \\ f[s] &:= \lambda n \, . \, s_i(n) \end{aligned}$$

Define the sequential algorithm $\{s_i\}_{i \in \mathbb{N}}$ by $s_0(n) = (*)$ (i.e. $M[s_0] = X$) and

1. set $n_i, p_i := \omega(M[s_i], f[s_i]), \phi(M[s_i], f[s_i])$
2. search from 0 up to n_i for some n such that
    - $x_n \in M[s_i]$
    - $\neg R_{[M[s_i]](n) \cup \{x_n\}}(p_i)$

1. if none is found, terminate
2. else, set $s_{i+1} := [s_i](n) :: p_i :: \lambda k \, . \, (*)$

Theorem

Whenever the above algorithm is run on continuous parameters (ω, ϕ), it terminates in some state s_j where $M[s_j], f[s_j]$ form an approximate maximal object w.r.t. (ω, ϕ).

The algorithm (corresponds to finding some F such that $\forall \phi P(F, \phi F)$)

A state s is defined to be a function $\mathbb{N} \to \{(*)\} + \mathbb{N}$. Any state induces a set $M[s]$ and a function $f : \mathrm{dom}(X \backslash M[s]) \to \mathbb{N}$ via

$$\begin{aligned} M[s] &:= \{x_n \mid s_i(n) = (*)\} \\ f[s] &:= \lambda n \ . \ s_i(n) \end{aligned}$$

Define the sequential algorithm $\{s_i\}_{i \in \mathbb{N}}$ by $s_0(n) = (*)$ (i.e. $M[s_0] = X$) and

1. set $n_i, p_i := \omega(M[s_i], f[s_i]), \phi(M[s_i], f[s_i])$
2. search from 0 up to n_i for some n such that
   - $x_n \in M[s_i]$
   - $\neg R_{[M[s_i]](n) \cup \{x_n\}}(p_i)$
3. if none is found, terminate
4. else, set $s_{i+1} := [s_i](n) :: p_i :: \lambda k \ . \ (*)$

Theorem

Whenever the above algorithm is run on continuous parameters (ω, ϕ), it terminates in some state s_j where $M[s_j], f[s_j]$ form an approximate maximal object w.r.t. (ω, ϕ).

The algorithm (corresponds to finding some F such that $\forall \phi P(F, \phi F)$)

A state s is defined to be a function $\mathbb{N} \to \{(*)\} + \mathbb{N}$. Any state induces a set $M[s]$ and a function $f : \mathrm{dom}(X \backslash M[s]) \to \mathbb{N}$ via

$$
\begin{aligned}
M[s] &:= \{x_n \mid s_i(n) = (*)\} \\
f[s] &:= \lambda n \, . \, s_i(n)
\end{aligned}
$$

Define the sequential algorithm $\{s_i\}_{i \in \mathbb{N}}$ by $s_0(n) = (*)$ (i.e. $M[s_0] = X$) and

1. set $n_i, p_i := \omega(M[s_i], f[s_i]), \phi(M[s_i], f[s_i])$
2. search from 0 up to n_i for some n such that
   - $x_n \in M[s_i]$
   - $\neg R_{[M[s_i]](n) \cup \{x_n\}}(p_i)$

1. if none is found, terminate
2. else, set $s_{i+1} := [s_i](n) :: p_i :: \lambda k \, . \, (*)$

Theorem

Whenever the above algorithm is run on continuous parameters (ω, ϕ), it terminates in some state s_j where $M[s_j], f[s_j]$ form an approximate maximal object w.r.t. (ω, ϕ).

The algorithm (corresponds to finding some F such that $\forall \phi P(F, \phi F)$)

A state s is defined to be a function $\mathbb{N} \to \{(*)\} + \mathbb{N}$. Any state induces a set $M[s]$ and a function $f : \mathrm{dom}(X \backslash M[s]) \to \mathbb{N}$ via

$$\begin{aligned} M[s] &:= \{x_n \mid s_i(n) = (*)\} \\ f[s] &:= \lambda n \,.\, s_i(n) \end{aligned}$$

Define the sequential algorithm $\{s_i\}_{i \in \mathbb{N}}$ by $s_0(n) = (*)$ (i.e. $M[s_0] = X$) and

1. set $n_i, p_i := \omega(M[s_i], f[s_i]), \phi(M[s_i], f[s_i])$
2. search from 0 up to n_i for some n such that
   - $x_n \in M[s_i]$
   - $\neg R_{[M[s_i]](n) \cup \{x_n\}}(p_i)$

1. if none is found, terminate
2. else, set $s_{i+1} := [s_i](n) :: p_i :: \lambda k \,.\, (*)$

Theorem

Whenever the above algorithm is run on continuous parameters (ω, ϕ), it terminates in some state s_j where $M[s_j], f[s_j]$ form an approximate maximal object w.r.t. (ω, ϕ).

The algorithm (corresponds to finding some F such that $\forall\phi P(F, \phi F)$)

A state s is defined to be a function $\mathbb{N} \to \{(*)\} + \mathbb{N}$. Any state induces a set $M[s]$ and a function $f : \mathrm{dom}(X \backslash M[s]) \to \mathbb{N}$ via

$$M[s] := \{x_n \mid s_i(n) = (*)\}$$
$$f[s] := \lambda n \, . \, s_i(n)$$

Define the sequential algorithm $\{s_i\}_{i\in\mathbb{N}}$ by $s_0(n) = (*)$ (i.e. $M[s_0] = X$) and

1. set $n_i, p_i := \omega(M[s_i], f[s_i]), \phi(M[s_i], f[s_i])$
2. search from 0 up to n_i for some n such that
    - $x_n \in M[s_i]$
    - $\neg R_{[M[s_i]](n)\cup\{x_n\}}(p_i)$

- if none is found, terminate
- else, set $s_{i+1} := [s_i](n) :: p_i :: \lambda k \, . \, (*)$

Theorem

Whenever the above algorithm is run on continuous parameters (ω, ϕ), it terminates in some state s_j where $M[s_j], f[s_j]$ form an approximate maximal object w.r.t. (ω, ϕ).

Application (corresponds to finding g, h such that $\forall x(P(x, gx) \to Q(hx))$)

It is well known that in any commutative ring:

$$r \text{ lies in intersection of all prime ideals} \Rightarrow r \text{ is nilpotent}$$

We can formalise this using $\rhd$ as before and $Q(x) := (\forall e > 0)(x \neq r^e)$.

Suppose $\psi : \mathcal{P}(X) \to \{0, 1, 2\} + (\{3, 4, 5\} \times \mathbb{N}^3)$ witnesses the premise of the above in the following sense:

- $\psi(S) = 0 \Rightarrow 0_X \notin S$
- $\psi(S) = 1 \Rightarrow 1_X \in S$
- $\psi(S) = 2 \Rightarrow r \in S$
- $\psi(S) = (3, i, j, k) \Rightarrow (x_i + x_j = x_k) \wedge (x_i, x_j \in S) \wedge (x_k \notin S)$
- $\psi(S) = (4, i, j, k) \Rightarrow (x_i \cdot x_j = x_k) \wedge (x_i \in S) \wedge (x_k \notin S)$
- $\psi(S) = (5, i, j, k) \Rightarrow (x_i \cdot x_j = x_k) \wedge (x_i, x_j \notin S) \wedge (x_k \in S)$

Then running our algorithm on suitable (ω_ψ, ϕ_ψ) defined in terms of ψ given us a way of computing an exponent $e > 0$ such that $r^e = 0$.

Application (corresponds to finding g, h such that $\forall x(P(x, gx) \to Q(hx))$)

It is well known that in any commutative ring:

$$r \text{ lies in intersection of all prime ideals} \Rightarrow r \text{ is nilpotent}$$

We can formalise this using $\rhd$ as before and $Q(x) := (\forall e > 0)(x \neq r^e)$.

Suppose $\psi : \mathcal{P}(X) \to \{0, 1, 2\} + (\{3, 4, 5\} \times \mathbb{N}^3)$ witnesses the premise of the above in the following sense:

- $\psi(S) = 0 \Rightarrow 0_X \notin S$
- $\psi(S) = 1 \Rightarrow 1_X \in S$
- $\psi(S) = 2 \Rightarrow r \in S$
- $\psi(S) = (3, i, j, k) \Rightarrow (x_i + x_j = x_k) \wedge (x_i, x_j \in S) \wedge (x_k \notin S)$
- $\psi(S) = (4, i, j, k) \Rightarrow (x_i \cdot x_j = x_k) \wedge (x_i \in S) \wedge (x_k \notin S)$
- $\psi(S) = (5, i, j, k) \Rightarrow (x_i \cdot x_j = x_k) \wedge (x_i, x_j \notin S) \wedge (x_k \in S)$

Then running our algorithm on suitable (ω_ψ, ϕ_ψ) defined in terms of ψ given us a way of computing an exponent $e > 0$ such that $r^e = 0$.

Application (corresponds to finding g, h such that $\forall x(P(x, gx) \to Q(hx))$)

It is well known that in any commutative ring:

$$r \text{ lies in intersection of all prime ideals} \Rightarrow r \text{ is nilpotent}$$

We can formalise this using $\rhd$ as before and $Q(x) := (\forall e > 0)(x \neq r^e)$.

Suppose $\psi : \mathcal{P}(X) \to \{0, 1, 2\} + (\{3, 4, 5\} \times \mathbb{N}^3)$ witnesses the premise of the above in the following sense:

- $\psi(S) = 0 \Rightarrow 0_X \notin S$
- $\psi(S) = 1 \Rightarrow 1_X \in S$
- $\psi(S) = 2 \Rightarrow r \in S$
- $\psi(S) = (3, i, j, k) \Rightarrow (x_i + x_j = x_k) \wedge (x_i, x_j \in S) \wedge (x_k \notin S)$
- $\psi(S) = (4, i, j, k) \Rightarrow (x_i \cdot x_j = x_k) \wedge (x_i \in S) \wedge (x_k \notin S)$
- $\psi(S) = (5, i, j, k) \Rightarrow (x_i \cdot x_j = x_k) \wedge (x_i, x_j \notin S) \wedge (x_k \in S)$

Then running our algorithm on suitable (ω_ψ, ϕ_ψ) defined in terms of ψ given us a way of computing an exponent $e > 0$ such that $r^e = 0$.

Application (corresponds to finding g, h such that $\forall x(P(x, gx) \to Q(hx))$)

It is well known that in any commutative ring:

$$r \text{ lies in intersection of all prime ideals} \Rightarrow r \text{ is nilpotent}$$

We can formalise this using $\rhd$ as before and $Q(x) := (\forall e > 0)(x \neq r^e)$.

Suppose $\psi : \mathcal{P}(X) \to \{0, 1, 2\} + (\{3, 4, 5\} \times \mathbb{N}^3)$ witnesses the premise of the above in the following sense:

- $\psi(S) = 0 \Rightarrow 0_X \notin S$
- $\psi(S) = 1 \Rightarrow 1_X \in S$
- $\psi(S) = 2 \Rightarrow r \in S$
- $\psi(S) = (3, i, j, k) \Rightarrow (x_i + x_j = x_k) \wedge (x_i, x_j \in S) \wedge (x_k \notin S)$
- $\psi(S) = (4, i, j, k) \Rightarrow (x_i \cdot x_j = x_k) \wedge (x_i \in S) \wedge (x_k \notin S)$
- $\psi(S) = (5, i, j, k) \Rightarrow (x_i \cdot x_j = x_k) \wedge (x_i, x_j \notin S) \wedge (x_k \in S)$

Then running our algorithm on suitable (ω_ψ, ϕ_ψ) defined in terms of ψ given us a way of computing an exponent $e > 0$ such that $r^e = 0$.

Informal description of algorithm in this case

- Each state s_i encodes some $M[s_i] \subseteq X$, where $x_n \notin M[s_i]$ only if we have found evidence that $[M[s_i]](n) \cup \{x_n\}$ generates r^e for some $e > 0$, in which case this evidence is encoded as $s_i(n) \in \mathbb{N}$.

- We start off at s_0 with the full set $M[s_0] = X$.

- At state s_i we interact with our functional ψ, which provides us with evidence that either $M[s_i]$ is not a prime ideal, or $r \in M[s_i]$.

- If this evidence takes the form of anything other than $0_X \notin S$, then we are able to use this to find some $x_n \in M$ and evidence that $[M](n) \cup \{x_n\}$ generates r^e for some $e > 0$. We exclude x_n from $M[s_i]$ but add all x_k for all $k > n$ (since now the evidence that $[M[s_i]](k) \cup \{x_k\}$ generates $r^{e'}$ could be falsified by the removal of x_n).

- Eventually, using a continuity argument, the algorithm terminates in some state s_j. The only way this can be is if $\psi(M[s_j]) = 0$, which indicates that $0_X \notin M[s_j]$. Thus $\{0_X\}$ generates r^e for some $e > 0$ encoded in the state.

Informal description of algorithm in this case

- Each state s_i encodes some $M[s_i] \subseteq X$, where $x_n \notin M[s_i]$ only if we have found evidence that $[M[s_i]](n) \cup \{x_n\}$ generates r^e for some $e > 0$, in which case this evidence is encoded as $s_i(n) \in \mathbb{N}$.
- We start off at s_0 with the full set $M[s_0] = X$.
- At state s_i we interact with our functional ψ, which provides us with evidence that either $M[s_i]$ is not a prime ideal, or $r \in M[s_i]$.
- If this evidence takes the form of anything other than $0_X \notin S$, then we are able to use this to find some $x_n \in M$ and evidence that $[M](n) \cup \{x_n\}$ generates r^e for some $e > 0$. We exclude x_n from $M[s_i]$ but add all x_k for all $k > n$ (since now the evidence that $[M[s_i]](k) \cup \{x_k\}$ generates $r^{e'}$ could be falsified by the removal of x_n).
- Eventually, using a continuity argument, the algorithm terminates in some state s_j. The only way this can be is if $\psi(M[s_j]) = 0$, which indicates that $0_X \notin M[s_j]$. Thus $\{0_X\}$ generates r^e for some $e > 0$ encoded in the state.

Informal description of algorithm in this case

- Each state s_i encodes some $M[s_i] \subseteq X$, where $x_n \notin M[s_i]$ only if we have found evidence that $[M[s_i]](n) \cup \{x_n\}$ generates r^e for some $e > 0$, in which case this evidence is encoded as $s_i(n) \in \mathbb{N}$.
- We start off at s_0 with the full set $M[s_0] = X$.
- At state s_i we interact with our functional ψ, which provides us with evidence that either $M[s_i]$ is not a prime ideal, or $r \in M[s_i]$.
- If this evidence takes the form of anything other than $0_X \notin S$, then we are able to use this to find some $x_n \in M$ and evidence that $[M](n) \cup \{x_n\}$ generates r^e for some $e > 0$. We exclude x_n from $M[s_i]$ but add all x_k for all $k > n$ (since now the evidence that $[M[s_i]](k) \cup \{x_k\}$ generates $r^{e'}$ could be falsified by the removal of x_n).
- Eventually, using a continuity argument, the algorithm terminates in some state s_j. The only way this can be is if $\psi(M[s_j]) = 0$, which indicates that $0_X \notin M[s_j]$. Thus $\{0_X\}$ generates r^e for some $e > 0$ encoded in the state.

Informal description of algorithm in this case

- Each state s_i encodes some $M[s_i] \subseteq X$, where $x_n \notin M[s_i]$ only if we have found evidence that $[M[s_i]](n) \cup \{x_n\}$ generates r^e for some $e > 0$, in which case this evidence is encoded as $s_i(n) \in \mathbb{N}$.

- We start off at s_0 with the full set $M[s_0] = X$.

- At state s_i we interact with our functional ψ, which provides us with evidence that either $M[s_i]$ is not a prime ideal, or $r \in M[s_i]$.

- If this evidence takes the form of anything other than $0_X \notin S$, then we are able to use this to find some $x_n \in M$ and evidence that $[M](n) \cup \{x_n\}$ generates r^e for some $e > 0$. We exclude x_n from $M[s_i]$ but add all x_k for all $k > n$ (since now the evidence that $[M[s_i]](k) \cup \{x_k\}$ generates $r^{e'}$ could be falsified by the removal of x_n).

- Eventually, using a continuity argument, the algorithm terminates in some state s_j. The only way this can be is if $\psi(M[s_j]) = 0$, which indicates that $0_X \notin M[s_j]$. Thus $\{0_X\}$ generates r^e for some $e > 0$ encoded in the state.

Informal description of algorithm in this case

- Each state s_i encodes some $M[s_i] \subseteq X$, where $x_n \notin M[s_i]$ only if we have found evidence that $[M[s_i]](n) \cup \{x_n\}$ generates r^e for some $e > 0$, in which case this evidence is encoded as $s_i(n) \in \mathbb{N}$.

- We start off at s_0 with the full set $M[s_0] = X$.

- At state s_i we interact with our functional ψ, which provides us with evidence that either $M[s_i]$ is not a prime ideal, or $r \in M[s_i]$.

- If this evidence takes the form of anything other than $0_X \notin S$, then we are able to use this to find some $x_n \in M$ and evidence that $[M](n) \cup \{x_n\}$ generates r^e for some $e > 0$. We exclude x_n from $M[s_i]$ but add all x_k for all $k > n$ (since now the evidence that $[M[s_i]](k) \cup \{x_k\}$ generates $r^{e'}$ could be falsified by the removal of x_n).

- Eventually, using a continuity argument, the algorithm terminates in some state s_j. The only way this can be is if $\psi(M[s_j]) = 0$, which indicates that $0_X \notin M[s_j]$. Thus $\{0_X\}$ generates r^e for some $e > 0$ encoded in the state.

Informal description of algorithm in this case

- Each state s_i encodes some $M[s_i] \subseteq X$, where $x_n \notin M[s_i]$ only if we have found evidence that $[M[s_i]](n) \cup \{x_n\}$ generates r^e for some $e > 0$, in which case this evidence is encoded as $s_i(n) \in \mathbb{N}$.
- We start off at s_0 with the full set $M[s_0] = X$.
- At state s_i we interact with our functional ψ, which provides us with evidence that either $M[s_i]$ is not a prime ideal, or $r \in M[s_i]$.
- If this evidence takes the form of anything other than $0_X \notin S$, then we are able to use this to find some $x_n \in M$ and evidence that $[M](n) \cup \{x_n\}$ generates r^e for some $e > 0$. We exclude x_n from $M[s_i]$ but add all x_k for all $k > n$ (since now the evidence that $[M[s_i]](k) \cup \{x_k\}$ generates $r^{e'}$ could be falsified by the removal of x_n).
- Eventually, using a continuity argument, the algorithm terminates in some state s_j. The only way this can be is if $\psi(M[s_j]) = 0$, which indicates that $0_X \notin M[s_j]$. Thus $\{0_X\}$ generates r^e for some $e > 0$ encoded in the state.

Concrete existential theorem: Nilpotent coefficients of invertible polynomials

We have achieved the following:

$$\underbrace{r \text{ lies in intersection of all prime ideals}}_{\text{witnessing functional } \psi} \quad \Rightarrow \quad \underbrace{r \text{ is nilpotent}}_{e > 0 \text{ with } r^e = 0}$$

For existential statements which use this, we simply need to instantiate ψ.

Theorem

Let $f = \sum_{i=0}^{n} a_i T^i$ be a unit in $X[T]$. Then a_i is nilpotent for all $i > 0$.

Proof.

Let $fg = 1$ and take some prime ideal P. Then $fg = 1$ also in $X/P[T]$, and since X/P is a domain we have $\deg(f) + \deg(g) = \deg(fg) = 0$. Thus for all $i > 0$, $a_i = 0$ in X/P and hence $a_i \in P$. Since a_i in intersection of all prime ideals, it is nilpotent. □

We can easily read off a suitable ψ from the above proof (but I won't give the messy details here!)

Concrete existential theorem: Nilpotent coefficients of invertible polynomials

We have achieved the following:

$$\underbrace{r \text{ lies in intersection of all prime ideals}}_{\text{witnessing functional } \psi} \quad \Rightarrow \quad \underbrace{r \text{ is nilpotent}}_{e > 0 \text{ with } r^e = 0}$$

For existential statements which use this, we simply need to instantiate ψ.

Theorem

Let $f = \sum_{i=0}^{n} a_i T^i$ be a unit in $X[T]$. Then a_i is nilpotent for all $i > 0$.

Proof.

Let $fg = 1$ and take some prime ideal P. Then $fg = 1$ also in $X/P[T]$, and since X/P is a domain we have $\deg(f) + \deg(g) = \deg(fg) = 0$. Thus for all $i > 0$, $a_i = 0$ in X/P and hence $a_i \in P$. Since a_i in intersection of all prime ideals, it is nilpotent. □

We can easily read off a suitable ψ from the above proof (but I won't give the messy details here!)

Concrete existential theorem: Nilpotent coefficients of invertible polynomials

We have achieved the following:

$$\underbrace{r \text{ lies in intersection of all prime ideals}}_{\text{witnessing functional } \psi} \quad \Rightarrow \quad \underbrace{r \text{ is nilpotent}}_{e > 0 \text{ with } r^e = 0}$$

For existential statements which use this, we simply need to instantiate ψ.

Theorem

Let $f = \sum_{i=0}^{n} a_i T^i$ be a unit in $X[T]$. Then a_i is nilpotent for all $i > 0$.

Proof.

Let $fg = 1$ and take some prime ideal P. Then $fg = 1$ also in $X/P[T]$, and since X/P is a domain we have $\deg(f) + \deg(g) = \deg(fg) = 0$. Thus for all $i > 0$, $a_i = 0$ in X/P and hence $a_i \in P$. Since a_i in intersection of all prime ideals, it is nilpotent. □

We can easily read off a suitable ψ from the above proof (but I won't give the messy details here!)

Concrete existential theorem: Nilpotent coefficients of invertible polynomials

We have achieved the following:

$$\underbrace{r \text{ lies in intersection of all prime ideals}}_{\text{witnessing functional } \psi} \quad \Rightarrow \quad \underbrace{r \text{ is nilpotent}}_{e > 0 \text{ with } r^e = 0}$$

For existential statements which use this, we simply need to instantiate ψ.

Theorem

Let $f = \sum_{i=0}^{n} a_i T^i$ be a unit in $X[T]$. Then a_i is nilpotent for all $i > 0$.

Proof.

Let $fg = 1$ and take some prime ideal P. Then $fg = 1$ also in $X/P[T]$, and since X/P is a domain we have $\deg(f) + \deg(g) = \deg(fg) = 0$. Thus for all $i > 0$, $a_i = 0$ in X/P and hence $a_i \in P$. Since a_i in intersection of all prime ideals, it is nilpotent. □

We can easily read off a suitable ψ from the above proof (but I won't give the messy details here!)

Concrete existential theorem: Nilpotent coefficients of invertible polynomials

We have achieved the following:

$$\underbrace{r \text{ lies in intersection of all prime ideals}}_{\text{witnessing functional } \psi} \quad \Rightarrow \quad \underbrace{r \text{ is nilpotent}}_{e > 0 \text{ with } r^e = 0}$$

For existential statements which use this, we simply need to instantiate ψ.

Theorem

Let $f = \sum_{i=0}^{n} a_i T^i$ be a unit in $X[T]$. Then a_i is nilpotent for all $i > 0$.

Proof.

Let $fg = 1$ and take some prime ideal P. Then $fg = 1$ also in $X/P[T]$, and since X/P is a domain we have $\deg(f) + \deg(g) = \deg(fg) = 0$. Thus for all $i > 0$, $a_i = 0$ in X/P and hence $a_i \in P$. Since a_i in intersection of all prime ideals, it is nilpotent. □

We can easily read off a suitable ψ from the above proof (but I won't give the messy details here!)

Concrete existential theorem: Nilpotent coefficients of invertible polynomials

We have achieved the following:

$$\underbrace{r \text{ lies in intersection of all prime ideals}}_{\text{witnessing functional } \psi} \quad \Rightarrow \quad \underbrace{r \text{ is nilpotent}}_{e>0 \text{ with } r^e=0}$$

For existential statements which use this, we simply need to instantiate ψ.

Theorem

Let $f = \sum_{i=0}^{n} a_i T^i$ be a unit in $X[T]$. Then a_i is nilpotent for all $i > 0$.

Proof.

Let $fg = 1$ and take some prime ideal P. Then $fg = 1$ also in $X/P[T]$, and since X/P is a domain we have $\deg(f) + \deg(g) = \deg(fg) = 0$. Thus for all $i > 0$, $a_i = 0$ in X/P and hence $a_i \in P$. Since a_i in intersection of all prime ideals, it is nilpotent. □

We can easily read off a suitable ψ from the above proof (but I won't give the messy details here!)

A (silly) example of a computation

Let $X = \mathbb{Z}_4$ and consider $f = 1 + 2T$, which is a unit in $\mathbb{Z}_4[T]$ since

$$(1 + 2T)(1 + 2T) = 1.$$

Then it follows that 2 is nilpotent in $\mathbb{Z}_4$.

Our algorithm would give rise to the following computation sequence:

$$\begin{aligned} s_0 = [(*), (*), (*), (*)] &\mapsto [(*), ([2], 1), (*), (*)] \\ &\mapsto [(*), ([1], 1), ([0, 1], 1), (*)] \\ &\mapsto [([0], 2), (*), (*), (*)] \end{aligned}$$

This corresponds to the following sequence of approximately maximal ideals

$$\mathbb{Z}_4 \mapsto \mathbb{Z}_4 \backslash \{1\} \mapsto \mathbb{Z}_4 \backslash \{1, 2\} \mapsto \mathbb{Z}_4 \backslash \{0\}$$

where the removal of an element is justified by including evidence that it can be used to generate 2^e for some $e > 0$.

The final state contains the information we need, namely $2^e = 0$ for $e = 2$.

A (silly) example of a computation

Let $X = \mathbb{Z}_4$ and consider $f = 1 + 2T$, which is a unit in $\mathbb{Z}_4[T]$ since

$$(1 + 2T)(1 + 2T) = 1.$$

Then it follows that 2 is nilpotent in $\mathbb{Z}_4$.

Our algorithm would give rise to the following computation sequence:

$$\begin{aligned} s_0 = [(*), (*), (*), (*)] &\mapsto [(*), ([2], 1), (*), (*)] \\ &\mapsto [(*), ([1], 1), ([0, 1], 1), (*)] \\ &\mapsto [([0], 2), (*), (*), (*)] \end{aligned}$$

This corresponds to the following sequence of approximately maximal ideals

$$\mathbb{Z}_4 \mapsto \mathbb{Z}_4 \backslash \{1\} \mapsto \mathbb{Z}_4 \backslash \{1, 2\} \mapsto \mathbb{Z}_4 \backslash \{0\}$$

where the removal of an element is justified by including evidence that it can be used to generate 2^e for some $e > 0$.

The final state contains the information we need, namely $2^e = 0$ for $e = 2$.

A (silly) example of a computation

Let $X = \mathbb{Z}_4$ and consider $f = 1 + 2T$, which is a unit in $\mathbb{Z}_4[T]$ since

$$(1 + 2T)(1 + 2T) = 1.$$

Then it follows that 2 is nilpotent in $\mathbb{Z}_4$.

Our algorithm would give rise to the following computation sequence:

$$\begin{aligned} s_0 = [(*),(*),(*),(*)] &\mapsto [(*),([2],1),(*),(*)] \\ &\mapsto [(*),([1],1),([0,1],1),(*)] \\ &\mapsto [([0],2),(*),(*),(*)] \end{aligned}$$

This corresponds to the following sequence of approximately maximal ideals

$$\mathbb{Z}_4 \mapsto \mathbb{Z}_4 \backslash \{1\} \mapsto \mathbb{Z}_4 \backslash \{1, 2\} \mapsto \mathbb{Z}_4 \backslash \{0\}$$

where the removal of an element is justified by including evidence that it can be used to generate 2^e for some $e > 0$.

The final state contains the information we need, namely $2^e = 0$ for $e = 2$.

A (silly) example of a computation

Let $X = \mathbb{Z}_4$ and consider $f = 1 + 2T$, which is a unit in $\mathbb{Z}_4[T]$ since

$$(1 + 2T)(1 + 2T) = 1.$$

Then it follows that 2 is nilpotent in $\mathbb{Z}_4$.

Our algorithm would give rise to the following computation sequence:

$$\begin{aligned} s_0 = [(*), (*), (*), (*)] &\mapsto [(*), ([2], 1), (*), (*)] \\ &\mapsto [(*), ([1], 1), ([0, 1], 1), (*)] \\ &\mapsto [([0], 2), (*), (*), (*)] \end{aligned}$$

This corresponds to the following sequence of approximately maximal ideals

$$\mathbb{Z}_4 \mapsto \mathbb{Z}_4 \backslash \{1\} \mapsto \mathbb{Z}_4 \backslash \{1, 2\} \mapsto \mathbb{Z}_4 \backslash \{0\}$$

where the removal of an element is justified by including evidence that it can be used to generate 2^e for some $e > 0$.

The final state contains the information we need, namely $2^e = 0$ for $e = 2$.

A (silly) example of a computation

Let $X = \mathbb{Z}_4$ and consider $f = 1 + 2T$, which is a unit in $\mathbb{Z}_4[T]$ since

$$(1 + 2T)(1 + 2T) = 1.$$

Then it follows that 2 is nilpotent in $\mathbb{Z}_4$.

Our algorithm would give rise to the following computation sequence:

$$\begin{aligned} s_0 = [(*), (*), (*), (*)] &\mapsto [(*), ([2], 1), (*), (*)] \\ &\mapsto [(*), ([1], 1), ([0, 1], 1), (*)] \\ &\mapsto [([0], 2), (*), (*), (*)] \end{aligned}$$

This corresponds to the following sequence of approximately maximal ideals

$$\mathbb{Z}_4 \mapsto \mathbb{Z}_4 \backslash \{1\} \mapsto \mathbb{Z}_4 \backslash \{1, 2\} \mapsto \mathbb{Z}_4 \backslash \{0\}$$

where the removal of an element is justified by including evidence that it can be used to generate 2^e for some $e > 0$.

The final state contains the information we need, namely $2^e = 0$ for $e = 2$.

A (silly) example of a computation

Let $X = \mathbb{Z}_4$ and consider $f = 1 + 2T$, which is a unit in $\mathbb{Z}_4[T]$ since

$$(1 + 2T)(1 + 2T) = 1.$$

Then it follows that 2 is nilpotent in $\mathbb{Z}_4$.

Our algorithm would give rise to the following computation sequence:

$$\begin{aligned} s_0 = [(*), (*), (*), (*)] &\mapsto [(*), ([2], 1), (*), (*)] \\ &\mapsto [(*), ([1], 1), ([0, 1], 1), (*)] \\ &\mapsto [([0], 2), (*), (*), (*)] \end{aligned}$$

This corresponds to the following sequence of approximately maximal ideals

$$\mathbb{Z}_4 \mapsto \mathbb{Z}_4 \backslash \{1\} \mapsto \mathbb{Z}_4 \backslash \{1, 2\} \mapsto \mathbb{Z}_4 \backslash \{0\}$$

where the removal of an element is justified by including evidence that it can be used to generate 2^e for some $e > 0$.

The final state contains the information we need, namely $2^e = 0$ for $e = 2$.

A (silly) example of a computation

Let $X = \mathbb{Z}_4$ and consider $f = 1 + 2T$, which is a unit in $\mathbb{Z}_4[T]$ since

$$(1 + 2T)(1 + 2T) = 1.$$

Then it follows that 2 is nilpotent in $\mathbb{Z}_4$.

Our algorithm would give rise to the following computation sequence:

$$\begin{aligned} s_0 = [(*), (*), (*), (*)] &\mapsto [(*), ([2], 1), (*), (*)] \\ &\mapsto [(*), ([1], 1), ([0, 1], 1), (*)] \\ &\mapsto [([0], 2), (*), (*), (*)] \end{aligned}$$

This corresponds to the following sequence of approximately maximal ideals

$$\mathbb{Z}_4 \mapsto \mathbb{Z}_4 \backslash \{1\} \mapsto \mathbb{Z}_4 \backslash \{1, 2\} \mapsto \mathbb{Z}_4 \backslash \{0\}$$

where the removal of an element is justified by including evidence that it can be used to generate 2^e for some $e > 0$.

The final state contains the information we need, namely $2^e = 0$ for $e = 2$.

A (silly) example of a computation

Let $X = \mathbb{Z}_4$ and consider $f = 1 + 2T$, which is a unit in $\mathbb{Z}_4[T]$ since

$$(1 + 2T)(1 + 2T) = 1.$$

Then it follows that 2 is nilpotent in $\mathbb{Z}_4$.

Our algorithm would give rise to the following computation sequence:

$$\begin{aligned} s_0 = [(*), (*), (*), (*)] &\mapsto [(*), ([2], 1), (*), (*)] \\ &\mapsto [(*), ([1], 1), ([0, 1], 1), (*)] \\ &\mapsto [([0], 2), (*), (*), (*)] \end{aligned}$$

This corresponds to the following sequence of approximately maximal ideals

$$\mathbb{Z}_4 \mapsto \mathbb{Z}_4 \setminus \{1\} \mapsto \mathbb{Z}_4 \setminus \{1, 2\} \mapsto \mathbb{Z}_4 \setminus \{0\}$$

where the removal of an element is justified by including evidence that it can be used to generate 2^e for some $e > 0$.

The final state contains the information we need, namely $2^e = 0$ for $e = 2$.

A (silly) example of a computation

Let $X = \mathbb{Z}_4$ and consider $f = 1 + 2T$, which is a unit in $\mathbb{Z}_4[T]$ since

$$(1 + 2T)(1 + 2T) = 1.$$

Then it follows that 2 is nilpotent in $\mathbb{Z}_4$.

Our algorithm would give rise to the following computation sequence:

$$\begin{aligned} s_0 = [(*), (*), (*), (*)] &\mapsto [(*), ([2], 1), (*), (*)] \\ &\mapsto [(*), ([1], 1), ([0, 1], 1), (*)] \\ &\mapsto [([0], 2), (*), (*), (*)] \end{aligned}$$

This corresponds to the following sequence of approximately maximal ideals

$$\mathbb{Z}_4 \mapsto \mathbb{Z}_4 \backslash \{1\} \mapsto \mathbb{Z}_4 \backslash \{1, 2\} \mapsto \mathbb{Z}_4 \backslash \{0\}$$

where the removal of an element is justified by including evidence that it can be used to generate 2^e for some $e > 0$.

The final state contains the information we need, namely $2^e = 0$ for $e = 2$.

A (silly) example of a computation

Let $X = \mathbb{Z}_4$ and consider $f = 1 + 2T$, which is a unit in $\mathbb{Z}_4[T]$ since

$$(1 + 2T)(1 + 2T) = 1.$$

Then it follows that 2 is nilpotent in $\mathbb{Z}_4$.

Our algorithm would give rise to the following computation sequence:

$$\begin{aligned} s_0 = [(*), (*), (*), (*)] &\mapsto [(*), ([2], 1), (*), (*)] \\ &\mapsto [(*), ([1], 1), ([0, 1], 1), (*)] \\ &\mapsto [([0], 2), (*), (*), (*)] \end{aligned}$$

This corresponds to the following sequence of approximately maximal ideals

$$\mathbb{Z}_4 \mapsto \mathbb{Z}_4 \backslash \{1\} \mapsto \mathbb{Z}_4 \backslash \{1, 2\} \mapsto \mathbb{Z}_4 \backslash \{0\}$$

where the removal of an element is justified by including evidence that it can be used to generate 2^e for some $e > 0$.

The final state contains the information we need, namely $2^e = 0$ for $e = 2$.

A (silly) example of a computation

Let $X = \mathbb{Z}_4$ and consider $f = 1 + 2T$, which is a unit in $\mathbb{Z}_4[T]$ since

$$(1 + 2T)(1 + 2T) = 1.$$

Then it follows that 2 is nilpotent in $\mathbb{Z}_4$.

Our algorithm would give rise to the following computation sequence:

$$\begin{aligned} s_0 = [(*), (*), (*), (*)] &\mapsto [(*), ([2], 1), (*), (*)] \\ &\mapsto [(*), ([1], 1), ([0, 1], 1), (*)] \\ &\mapsto [([0], 2), (*), (*), (*)] \end{aligned}$$

This corresponds to the following sequence of approximately maximal ideals

$$\mathbb{Z}_4 \mapsto \mathbb{Z}_4 \backslash \{1\} \mapsto \mathbb{Z}_4 \backslash \{1, 2\} \mapsto \mathbb{Z}_4 \backslash \{0\}$$

where the removal of an element is justified by including evidence that it can be used to generate 2^e for some $e > 0$.

The final state contains the information we need, namely $2^e = 0$ for $e = 2$.

A (silly) example of a computation

Let $X = \mathbb{Z}_4$ and consider $f = 1 + 2T$, which is a unit in $\mathbb{Z}_4[T]$ since

$$(1 + 2T)(1 + 2T) = 1.$$

Then it follows that 2 is nilpotent in $\mathbb{Z}_4$.

Our algorithm would give rise to the following computation sequence:

$$\begin{aligned} s_0 = [(*), (*), (*), (*)] &\mapsto [(*), ([2], 1), (*), (*)] \\ &\mapsto [(*), ([1], 1), ([0, 1], 1), (*)] \\ &\mapsto [([0], 2), (*), (*), (*)] \end{aligned}$$

This corresponds to the following sequence of approximately maximal ideals

$$\mathbb{Z}_4 \mapsto \mathbb{Z}_4 \backslash \{1\} \mapsto \mathbb{Z}_4 \backslash \{1, 2\} \mapsto \mathbb{Z}_4 \backslash \{0\}$$

where the removal of an element is justified by including evidence that it can be used to generate 2^e for some $e > 0$.

The final state contains the information we need, namely $2^e = 0$ for $e = 2$.

A (silly) example of a computation

Let $X = \mathbb{Z}_4$ and consider $f = 1 + 2T$, which is a unit in $\mathbb{Z}_4[T]$ since

$$(1 + 2T)(1 + 2T) = 1.$$

Then it follows that 2 is nilpotent in $\mathbb{Z}_4$.

Our algorithm would give rise to the following computation sequence:

$$\begin{aligned} s_0 = [(*), (*), (*), (*)] &\mapsto [(*), ([2], 1), (*), (*)] \\ &\mapsto [(*), ([1], 1), ([0, 1], 1), (*)] \\ &\mapsto [([0], 2), (*), (*), (*)] \end{aligned}$$

This corresponds to the following sequence of approximately maximal ideals

$$\mathbb{Z}_4 \mapsto \mathbb{Z}_4 \backslash \{1\} \mapsto \mathbb{Z}_4 \backslash \{1, 2\} \mapsto \mathbb{Z}_4 \backslash \{0\}$$

where the removal of an element is justified by including evidence that it can be used to generate 2^e for some $e > 0$.

The final state contains the information we need, namely $2^e = 0$ for $e = 2$.

A (silly) example of a computation

Let $X = \mathbb{Z}_4$ and consider $f = 1 + 2T$, which is a unit in $\mathbb{Z}_4[T]$ since

$$(1 + 2T)(1 + 2T) = 1.$$

Then it follows that 2 is nilpotent in $\mathbb{Z}_4$.

Our algorithm would give rise to the following computation sequence:

$$\begin{aligned} s_0 = [(*), (*), (*), (*)] &\mapsto [(*), ([2], 1), (*), (*)] \\ &\mapsto [(*), ([1], 1), ([0, 1], 1), (*)] \\ &\mapsto [([0], 2), (*), (*), (*)] \end{aligned}$$

This corresponds to the following sequence of approximately maximal ideals

$$\mathbb{Z}_4 \mapsto \mathbb{Z}_4 \backslash \{1\} \mapsto \mathbb{Z}_4 \backslash \{1, 2\} \mapsto \mathbb{Z}_4 \backslash \{0\}$$

where the removal of an element is justified by including evidence that it can be used to generate 2^e for some $e > 0$.

The final state contains the information we need, namely $2^e = 0$ for $e = 2$.

Another concrete existential theorem: Gauss-Joyal

Theorem

Take $f = \sum_{i=0}^{n} a_i T^i$ and $g = \sum_{i=0}^{m} b_i T^i$ in $X[T]$ and write $fg = \sum_{i=0}^{n+m} c_i T^i$. Then

$$a_i b_j \in \sqrt{(c_0, \ldots, c_{i+j})}$$

for all $i = 0, \ldots, n$ and $j = 0, \ldots, m$.

Proof (sketch).

We show that $a_i b_j \in P$ for all prime ideals with $\{c_0, \ldots, c_{i+j}\} \subseteq P$. Then $\{c_0, \ldots, c_{i+j}\}$ generates $(a_i b_j)^e$ for some $e > 0$, and thus $a_i b_j \in \sqrt{(c_0, \ldots, c_{i+j})}$. □

A generalisation of our framework enables us to produce, in a uniform way, a sequential algorithm which for any i, j computes some

- $x_0, \ldots, x_{i+j} \in X$,
- $e > 0$

such that $c_0 x_0 + \ldots + c_{i+j} x_{i+j} = (a_i b_j)^e$.

Another concrete existential theorem: Gauss-Joyal

Theorem

Take $f = \sum_{i=0}^{n} a_i T^i$ and $g = \sum_{i=0}^{m} b_i T^i$ in $X[T]$ and write $fg = \sum_{i=0}^{n+m} c_i T^i$. Then

$$a_i b_j \in \sqrt{(c_0, \ldots, c_{i+j})}$$

for all $i = 0, \ldots, n$ and $j = 0, \ldots, m$.

Proof (sketch).

We show that $a_i b_j \in P$ for all prime ideals with $\{c_0, \ldots, c_{i+j}\} \subseteq P$. Then $\{c_0, \ldots, c_{i+j}\}$ generates $(a_i b_j)^e$ for some $e > 0$, and thus $a_i b_j \in \sqrt{(c_0, \ldots, c_{i+j})}$. □

A generalisation of our framework enables us to produce, in a uniform way, a sequential algorithm which for any i, j computes some

- $x_0, \ldots, x_{i+j} \in X$,
- $e > 0$

such that $c_0 x_0 + \ldots + c_{i+j} x_{i+j} = (a_i b_j)^e$.

Another concrete existential theorem: Gauss-Joyal

Theorem

Take $f = \sum_{i=0}^{n} a_i T^i$ *and* $g = \sum_{i=0}^{m} b_i T^i$ *in* $X[T]$ *and write* $fg = \sum_{i=0}^{n+m} c_i T^i$. *Then*

$$a_i b_j \in \sqrt{(c_0, \ldots, c_{i+j})}$$

for all $i = 0, \ldots, n$ *and* $j = 0, \ldots, m$.

Proof (sketch).

We show that $a_i b_j \in P$ for all prime ideals with $\{c_0, \ldots, c_{i+j}\} \subseteq P$. Then $\{c_0, \ldots, c_{i+j}\}$ generates $(a_i b_j)^e$ for some $e > 0$, and thus $a_i b_j \in \sqrt{(c_0, \ldots, c_{i+j})}$. □

A generalisation of our framework enables us to produce, in a uniform way, a sequential algorithm which for any i, j computes some

- $x_0, \ldots, x_{i+j} \in X$,
- $e > 0$

such that $c_0 x_0 + \ldots + c_{i+j} x_{i+j} = (a_i b_j)^e$.

Another concrete existential theorem: Gauss-Joyal

Theorem

Take $f = \sum_{i=0}^{n} a_i T^i$ *and* $g = \sum_{i=0}^{m} b_i T^i$ *in* $X[T]$ *and write* $fg = \sum_{i=0}^{n+m} c_i T^i$. *Then*

$$a_i b_j \in \sqrt{(c_0, \ldots, c_{i+j})}$$

for all $i = 0, \ldots, n$ *and* $j = 0, \ldots, m$.

Proof (sketch).

We show that $a_i b_j \in P$ for all prime ideals with $\{c_0, \ldots, c_{i+j}\} \subseteq P$. Then $\{c_0, \ldots, c_{i+j}\}$ generates $(a_i b_j)^e$ for some $e > 0$, and thus $a_i b_j \in \sqrt{(c_0, \ldots, c_{i+j})}$. □

A generalisation of our framework enables us to produce, in a uniform way, a sequential algorithm which for any i, j computes some

- $x_0, \ldots, x_{i+j} \in X$,
- $e > 0$

such that $c_0 x_0 + \ldots + c_{i+j} x_{i+j} = (a_i b_j)^e$.

Conclusion and open questions

In this talk, I hope to have given some insight into how

(a) Gödel's functional interpretation

(b) sequential algorithms

can be used to construct witnesses for existential theorems in algebra.

There are a number of open problems:

1. How many interesting theorems can be dealt with in a uniform way through our main computational framework?
2. So far we assume that our underlying algebraic structure is countable. Can we generalise this, perhaps by introducing some 'abstract type' X for representing arbitrary commutative rings.
3. How do our algorithms compare to those which arise from dynamical algebra?

Conclusion and open questions

In this talk, I hope to have given some insight into how

(a) Gödel's functional interpretation

(b) sequential algorithms

can be used to construct witnesses for existential theorems in algebra.

There are a number of open problems:

1. How many interesting theorems can be dealt with in a uniform way through our main computational framework?
2. So far we assume that our underlying algebraic structure is countable. Can we generalise this, perhaps by introducing some 'abstract type' X for representing arbitrary commutative rings.
3. How do our algorithms compare to those which arise from dynamical algebra?

Conclusion and open questions

In this talk, I hope to have given some insight into how

(a) Gödel's functional interpretation

(b) sequential algorithms

can be used to construct witnesses for existential theorems in algebra.

There are a number of open problems:

1. How many interesting theorems can be dealt with in a uniform way through our main computational framework?
2. So far we assume that our underlying algebraic structure is countable. Can we generalise this, perhaps by introducing some 'abstract type' X for representing arbitrary commutative rings.
3. How do our algorithms compare to those which arise from dynamical algebra?

Conclusion and open questions

In this talk, I hope to have given some insight into how

(a) Gödel's functional interpretation

(b) sequential algorithms

can be used to construct witnesses for existential theorems in algebra.

There are a number of open problems:

1. How many interesting theorems can be dealt with in a uniform way through our main computational framework?
2. So far we assume that our underlying algebraic structure is countable. Can we generalise this, perhaps by introducing some 'abstract type' X for representing arbitrary commutative rings.
3. How do our algorithms compare to those which arise from dynamical algebra?

Conclusion and open questions

In this talk, I hope to have given some insight into how

(a) Gödel's functional interpretation

(b) sequential algorithms

can be used to construct witnesses for existential theorems in algebra.

There are a number of open problems:

1. How many interesting theorems can be dealt with in a uniform way through our main computational framework?
2. So far we assume that our underlying algebraic structure is countable. Can we generalise this, perhaps by introducing some 'abstract type' X for representing arbitrary commutative rings.
3. How do our algorithms compare to those which arise from dynamical algebra?

Conclusion and open questions

In this talk, I hope to have given some insight into how

(a) Gödel's functional interpretation

(b) sequential algorithms

can be used to construct witnesses for existential theorems in algebra.

There are a number of open problems:

1. How many interesting theorems can be dealt with in a uniform way through our main computational framework?
2. So far we assume that our underlying algebraic structure is countable. Can we generalise this, perhaps by introducing some 'abstract type' X for representing arbitrary commutative rings.
3. How do our algorithms compare to those which arise from dynamical algebra?

References

For more details, see our initial paper:

- *An algorithmic approach to the existence of ideal objects in commutative algebra*
  T. Powell, P. Schuster and F. Wiesnet. **Proceedings of WoLLIC '19**, LNCS 11541: 533–549, 2019.

An extended follow up paper encompassing a lot more is on the way:

- *A universal algorithm for Krull's lemma (working title)*
  T. Powell, P. Schuster and F. Wiesnet. **In progress**.

Our work uses ideas from the following in particular:

- *A universal Krull-Lindenbaum theorem*
  D. Rinaldi and P. Schuster. **Journal of Pure and Applied Algebra**, 200: 3207–3232, 2016.
- *Sequential algorithms and the computational content of classical proofs*
  T. Powell. **Preprint**, `https://arxiv.org/abs/1812.11003`, 2019.

Thank you!

References

For more details, see our initial paper:

- *An algorithmic approach to the existence of ideal objects in commutative algebra*
  T. Powell, P. Schuster and F. Wiesnet. **Proceedings of WoLLIC '19**, LNCS 11541: 533–549, 2019.

An extended follow up paper encompassing a lot more is on the way:

- *A universal algorithm for Krull's lemma (working title)*
  T. Powell, P. Schuster and F. Wiesnet. **In progress**.

Our work uses ideas from the following in particular:

- *A universal Krull-Lindenbaum theorem*
  D. Rinaldi and P. Schuster. **Journal of Pure and Applied Algebra**, 200: 3207–3232, 2016.
- *Sequential algorithms and the computational content of classical proofs*
  T. Powell. **Preprint**, `https://arxiv.org/abs/1812.11003`, 2019.

THANK YOU!

References

For more details, see our initial paper:

- *An algorithmic approach to the existence of ideal objects in commutative algebra*
  T. Powell, P. Schuster and F. Wiesnet. **Proceedings of WoLLIC '19**, LNCS 11541: 533–549, 2019.

An extended follow up paper encompassing a lot more is on the way:

- *A universal algorithm for Krull's lemma (working title)*
  T. Powell, P. Schuster and F. Wiesnet. **In progress**.

Our work uses ideas from the following in particular:

- *A universal Krull-Lindenbaum theorem*
  D. Rinaldi and P. Schuster. **Journal of Pure and Applied Algebra**, 200: 3207–3232, 2016.
- *Sequential algorithms and the computational content of classical proofs*
  T. Powell. **Preprint**, `https://arxiv.org/abs/1812.11003`, 2019.

THANK YOU!

References

For more details, see our initial paper:

- *An algorithmic approach to the existence of ideal objects in commutative algebra*
  T. Powell, P. Schuster and F. Wiesnet. **Proceedings of WoLLIC '19**, LNCS 11541: 533–549, 2019.

An extended follow up paper encompassing a lot more is on the way:

- *A universal algorithm for Krull's lemma (working title)*
  T. Powell, P. Schuster and F. Wiesnet. **In progress**.

Our work uses ideas from the following in particular:

- *A universal Krull-Lindenbaum theorem*
  D. Rinaldi and P. Schuster. **Journal of Pure and Applied Algebra**, 200: 3207–3232, 2016.
- *Sequential algorithms and the computational content of classical proofs*
  T. Powell. **Preprint**, `https://arxiv.org/abs/1812.11003`, 2019.

Thank you!

References

For more details, see our initial paper:

- *An algorithmic approach to the existence of ideal objects in commutative algebra*
  T. Powell, P. Schuster and F. Wiesnet. **Proceedings of WoLLIC '19**, LNCS 11541: 533–549, 2019.

An extended follow up paper encompassing a lot more is on the way:

- *A universal algorithm for Krull's lemma (working title)*
  T. Powell, P. Schuster and F. Wiesnet. **In progress**.

Our work uses ideas from the following in particular:

- *A universal Krull-Lindenbaum theorem*
  D. Rinaldi and P. Schuster. **Journal of Pure and Applied Algebra**, 200: 3207–3232, 2016.
- *Sequential algorithms and the computational content of classical proofs*
  T. Powell. **Preprint**, `https://arxiv.org/abs/1812.11003`, 2019.

Thank you!