

THE UNIVERSITY of EDINBURGH
informatics





Engineering and
Physical Sciences
Research Council



Program Synthesis by Semantics

Zoey Shepherd, supervised by Elizabeth Polgreen and Ohad Kammar

School of Informatics, University of Edinburgh

I want programs

I want programs that work

How do I get programs that work?



THE UNIVERSITY of EDINBURGH
informatics





Engineering and
Physical Sciences
Research Council

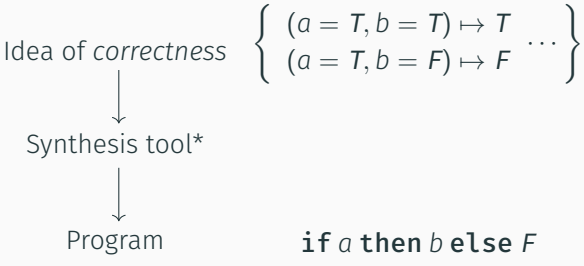


Program Synthesis by Semantics

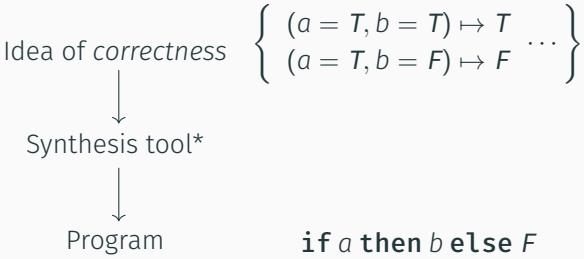
Zoey Shepherd, supervised by Elizabeth Polgreen and Ohad Kammar

School of Informatics, University of Edinburgh

Program Synthesis by Semantics



Program Synthesis by Semantics



*with these properties:

- **Soundness:** if the tool produces a program, the program is *correct*.
- **Completeness:** if a *correct* program could exist, the tool can produce it.

Idea of Correctness = Specification

- Examples:

$$\left\{ \begin{array}{ll} (a = T, b = T) \mapsto T & (a = T, b = F) \mapsto F \\ (a = F, b = T) \mapsto F & (a = F, b = F) \mapsto F \end{array} \right\} : \mathsf{Ex}[a : \mathbb{B}, b : \mathbb{B} \vdash \mathbb{B}]$$

Idea of Correctness = Specification

- Examples:

$$\left\{ \begin{array}{ll} (a = T, b = T) \mapsto T & (a = T, b = F) \mapsto F \\ (a = F, b = T) \mapsto F & (a = F, b = F) \mapsto F \end{array} \right\} : \mathsf{Ex}[a : \mathbb{B}, b : \mathbb{B} \vdash \mathbb{B}]$$

- Logic:

$$\begin{array}{rl} \forall a, b \in \mathbb{B}, & \exists f(a, b) \in \mathbb{B} \\ \text{s.t.} & (a \vee \neg f(a, b)) \wedge (b \vee \neg f(a, b)) \wedge (\neg a \vee \neg b \vee f(a, b)) \end{array}$$

Idea of Correctness = Specification

- Examples:

$$\left\{ \begin{array}{ll} (a = T, b = T) \mapsto T & (a = T, b = F) \mapsto F \\ (a = F, b = T) \mapsto F & (a = F, b = F) \mapsto F \end{array} \right\} : \mathsf{Ex}[a : \mathbb{B}, b : \mathbb{B} \vdash \mathbb{B}]$$

- Logic:

$$\forall a, b \in \mathbb{B}, \quad \exists f(a, b) \in \mathbb{B}$$
$$\text{s.t.} \quad (a \vee \neg f(a, b)) \wedge (b \vee \neg f(a, b)) \wedge (\neg a \vee \neg b \vee f(a, b))$$

- A reference implementation:

```
(..._)=>_[+[]]?_[+!+[]]:_[+[]]
```

Idea of Correctness = Specification

- Examples:

$$\left\{ \begin{array}{ll} (a = T, b = T) \mapsto T & (a = T, b = F) \mapsto F \\ (a = F, b = T) \mapsto F & (a = F, b = F) \mapsto F \end{array} \right\} : \mathsf{Ex}[a : \mathbb{B}, b : \mathbb{B} \vdash \mathbb{B}]$$

- Logic:

$$\forall a, b \in \mathbb{B}, \quad \exists f(a, b) \in \mathbb{B}$$
$$\text{s.t.} \quad (a \vee \neg f(a, b)) \wedge (b \vee \neg f(a, b)) \wedge (\neg a \vee \neg b \vee f(a, b))$$

- A reference implementation:

```
(..._)=>_[+[]]?_[+!+[]]:_[+[]]
```

- ...

Program = Term in a Language

- An actual programming language:

  Python, Lean, JavaScript, ...

Program = Term in a Language

- An actual programming language:

  Python, Lean, JavaScript, ...

- From a CFG:

$$\mathbb{B} ::= a \mid b \mid T \mid F \mid \texttt{if}\ \mathbb{B}\ \texttt{then}\ \mathbb{B}\ \texttt{else}\ \mathbb{B} \mid \texttt{not}\ \mathbb{B} \mid \ldots$$

$$\mathbb{N} ::= x \mid y \mid \mathbb{N}\ \texttt{+}\ \mathbb{N} \mid \mathbb{N}\ \texttt{*}\ \mathbb{N} \mid \texttt{if}\ \mathbb{B}\ \texttt{then}\ \mathbb{N}\ \texttt{else}\ \mathbb{N} \mid \ldots$$

Program = Term in a Language

- An actual programming language:

  Python, Lean, JavaScript, ...

- From a CFG:

$$\mathbb{B} \coloneqq a \mid b \mid T \mid F \mid \texttt{if}\ \mathbb{B}\ \texttt{then}\ \mathbb{B}\ \texttt{else}\ \mathbb{B} \mid \texttt{not}\ \mathbb{B} \mid \ldots$$

$$\mathbb{N} \coloneqq x \mid y \mid \mathbb{N} \texttt{+} \mathbb{N} \mid \mathbb{N} \texttt{*} \mathbb{N} \mid \texttt{if}\ \mathbb{B}\ \texttt{then}\ \mathbb{N}\ \texttt{else}\ \mathbb{N} \mid \ldots$$

- Multiset of operations:

$$2 : (-x_i) \quad 2 : (x_i + x_j) \quad 2 : (x_i - x_j) \quad 1 : (x_i * x_j)$$
$$3 : (!x_i) \quad 3 : (x_i \,\&\, x_j) \quad 3 : (x_i \mid x_j) \quad 3 : (x_i \oplus x_j)$$

Program = Term in a Language

- An actual programming language:

  Python, Lean, JavaScript, ...

- From a CFG:

$$\mathbb{B} \coloneqq a \mid b \mid T \mid F \mid \texttt{if}\ \mathbb{B}\ \texttt{then}\ \mathbb{B}\ \texttt{else}\ \mathbb{B} \mid \texttt{not}\ \mathbb{B} \mid \ldots$$

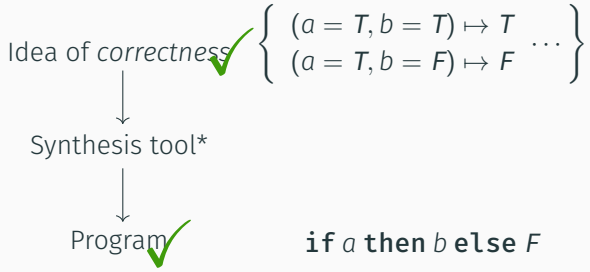
$$\mathbb{N} \coloneqq x \mid y \mid \mathbb{N}\ \texttt{+}\ \mathbb{N} \mid \mathbb{N}\ \texttt{*}\ \mathbb{N} \mid \texttt{if}\ \mathbb{B}\ \texttt{then}\ \mathbb{N}\ \texttt{else}\ \mathbb{N} \mid \ldots$$

- Multiset of operations:

$$2 : (-x_i) \quad 2 : (x_i + x_j) \quad 2 : (x_i - x_j) \quad 1 : (x_i * x_j)$$
$$3 : (!x_i) \quad 3 : (x_i \,\&\, x_j) \quad 3 : (x_i \mid x_j) \quad 3 : (x_i \oplus x_j)$$

- ...

Program Synthesis by Semantics



- **Soundness:** if the tool produces a program, the program is *correct*.
- **Completeness:** if a *correct* program could exist, the tool can produce it.

Synthesis Tool $\approx$ Search Procedure

- Bottom-up:

$$P_1^{\mathbb{B}} = \{T, F, a, b\}$$
$$P_2^{\mathbb{B}} = \{\textbf{not}\, t \mid t \in P_1^{\mathbb{B}}\}$$
$$P_3^{\mathbb{B}} = \{\textbf{not}\, t \mid t \in P_2^{\mathbb{B}}\} \cup \{t_1 \,\textbf{and}\, t_2 \mid t_1, t_2 \in P_1^{\mathbb{B}}\} \cup \{t_1 \,\textbf{or}\, t_2 \mid t_1, t_2 \in P_1^{\mathbb{B}}\} \cup \ldots$$

Synthesis Tool ≈ Search Procedure

- Bottom-up:

$$P_1^{\mathbb{B}} = \{T, F, a, b\}$$
$$P_2^{\mathbb{B}} = \{\mathbf{not}\, t \mid t \in P_1^{\mathbb{B}}\}$$
$$P_3^{\mathbb{B}} = \{\mathbf{not}\, t \mid t \in P_2^{\mathbb{B}}\} \cup \{t_1\, \mathbf{and}\, t_2 \mid t_1, t_2 \in P_1^{\mathbb{B}}\} \cup \{t_1\, \mathbf{or}\, t_2 \mid t_1, t_2 \in P_1^{\mathbb{B}}\} \cup \ldots$$

- Top-down:

$$\begin{aligned}[-] &= \mathbf{if}\,[-]\,\mathbf{then}\,[-]\,\mathbf{else}\,[-] \\ &= \mathbf{if}\,(\mathbf{not}\,[-])\,\mathbf{then}\,([-]\,\mathbf{+}\,[-])\,\mathbf{else}\,([-]\,\mathbf{*}\,[-])\end{aligned}$$

Synthesis Tool ≈ Search Procedure

- Bottom-up:

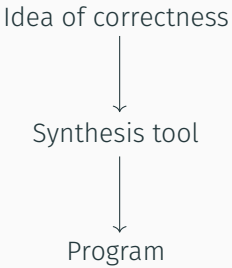
$$P_1^{\mathbb{B}} = \{T, F, a, b\}$$
$$P_2^{\mathbb{B}} = \{\mathbf{not}\ t \mid t \in P_1^{\mathbb{B}}\}$$
$$P_3^{\mathbb{B}} = \{\mathbf{not}\ t \mid t \in P_2^{\mathbb{B}}\} \cup \{t_1\ \mathbf{and}\ t_2 \mid t_1, t_2 \in P_1^{\mathbb{B}}\} \cup \{t_1\ \mathbf{or}\ t_2 \mid t_1, t_2 \in P_1^{\mathbb{B}}\} \cup \ldots$$

- Top-down:

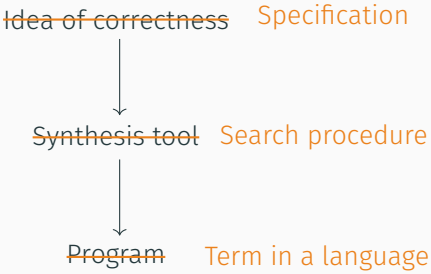
$$\begin{aligned}[-] &= \mathbf{if}\ [-]\ \mathbf{then}\ [-]\ \mathbf{else}\ [-] \\ &= \mathbf{if}\ (\mathbf{not}\ [-])\ \mathbf{then}\ ([-]\ \texttt{+}\ [-])\ \mathbf{else}\ ([-]\ \texttt{*}\ [-])\end{aligned}$$

- ...

Program Synthesis by Semantics



Program Synthesis by Semantics



Program Synthesis by Semantics

~~Idea of correctness~~ Specification $\left\{ \begin{array}{ll} (a = T, b = T) \mapsto T & (a = T, b = F) \mapsto F \\ (a = F, b = T) \mapsto F & (a = F, b = F) \mapsto F \end{array} \right\}$



~~Synthesis tool~~ Search procedure

$$P_1^{\mathbb{B}} = \{T, F, a, b\}$$
$$P_4^{\mathbb{B}} = \{\textbf{if}\ t_1\ \textbf{then}\ t_2\ \textbf{else}\ t_3 \mid t_1, t_2, t_3 \in P_1^{\mathbb{B}}\}$$
$$\vdots$$



~~Program~~ Term in a language $\mathbb{B} \coloneqq a \mid b \mid T \mid F \mid \textbf{if}\ \mathbb{B}\ \textbf{then}\ \mathbb{B}\ \textbf{else}\ \mathbb{B} \mid \ldots$

The haystack

T, F, a, b,

if T then T else T	if F then T else T	if a then T else T	if b then T else T
if T then T else F	if F then T else F	if a then T else F	if b then T else F
if T then T else a	if F then T else a	if a then T else a	if b then T else a
if T then T else b	if F then T else b	if a then T else b	if b then T else b
if T then F else T	if F then F else T	if a then F else T	if b then F else T
if T then F else F	if F then F else F	if a then F else F	if b then F else F
if T then F else a	if F then F else a	if a then F else a	if b then F else a
if T then F else b	if F then F else b	if a then F else b	if b then F else b
if T then a else T	if F then a else T	if a then a else T	if b then a else T
if T then a else F	if F then a else F	if a then a else F	if b then a else F
if T then a else a	if F then a else a	if a then a else a	if b then a else a
if T then a else b	if F then a else b	if a then a else b	if b then a else b
if T then b else T	if F then b else T	if a then b else T	if b then b else T
if T then b else F	if F then b else F	if a then b else F	if b then b else F
⋮	⋮	⋮	⋮

The haystack

T, F, a, b,

if T then T else T	if F then T else T	if a then T else T	if b then T else T
if T then T else F	if F then T else F	if a then T else F	if b then T else F
if T then T else a	if F then T else a	if a then T else a	if b then T else a
if T then T else b	if F then T else b	if a then T else b	if b then T else b
if T then F else T	if F then F else T	if a then F else T	if b then F else T
if T then F else F	if F then F else F	if a then F else F	if b then F else F
if T then F else a	if F then F else a	if a then F else a	if b then F else a
if T then F else b	if F then F else b	if a then F else b	if b then F else b
if T then a else T	if F then a else T	if a then a else T	if b then a else T
if T then a else F	if F then a else F	if a then a else F	if b then a else F
if T then a else a	if F then a else a	if a then a else a	if b then a else a
if T then a else b	if F then a else b	if a then a else b	if b then a else b
if T then b else T	if F then b else T	if a then b else T	if b then b else T
if T then b else F	if F then b else F	if a then b else F	if b then b else F
⋮	⋮	⋮	⋮

Program Synthesis by Semantics

- *Operational* semantics: What does the program do?

$$\frac{c \to^* T \quad t_1 \to^* v}{\texttt{if}\ c\ \texttt{then}\ t_1\ \texttt{else}\ t_2 \to^* v}$$

Program Synthesis by Semantics

- *Operational* semantics: What does the program do?

$$\frac{c \to^* T \quad t_1 \to^* v}{\texttt{if}\ c\ \texttt{then}\ t_1\ \texttt{else}\ t_2 \to^* v}$$

- *Equational* semantics: What does the program equal?

$$\texttt{if}\ T\ \texttt{then}\ t_1\ \texttt{else}\ t_2 \simeq t_1, \qquad \texttt{if}\ c\ \texttt{then}\ t\ \texttt{else}\ t \simeq t$$

Program Synthesis by Semantics

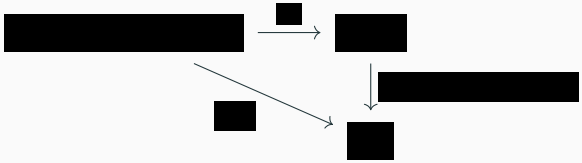
- *Operational* semantics: What does the program do?

$$\frac{c \to^* T \quad t_1 \to^* v}{\texttt{if}\ c\ \texttt{then}\ t_1\ \texttt{else}\ t_2 \to^* v}$$

- *Equational* semantics: What does the program equal?

$$\texttt{if}\ T\ \texttt{then}\ t_1\ \texttt{else}\ t_2 \simeq t_1, \qquad \texttt{if}\ c\ \texttt{then}\ t\ \texttt{else}\ t \simeq t$$

- *Denotational* semantics: What does the program denote?



The haystack

T, F, a, b,

if T then T else T	if F then T else T	if a then T else T	if b then T else T
if T then T else F	if F then T else F	if a then T else F	if b then T else F
if T then T else a	if F then T else a	if a then T else a	if b then T else a
if T then T else b	if F then T else b	if a then T else b	if b then T else b
if T then F else T	if F then F else T	if a then F else T	if b then F else T
if T then F else F	if F then F else F	if a then F else F	if b then F else F
if T then F else a	if F then F else a	if a then F else a	if b then F else a
if T then F else b	if F then F else b	if a then F else b	if b then F else b
if T then a else T	if F then a else T	if a then a else T	if b then a else T
if T then a else F	if F then a else F	if a then a else F	if b then a else F
if T then a else a	if F then a else a	if a then a else a	if b then a else a
if T then a else b	if F then a else b	if a then a else b	if b then a else b
if T then b else T	if F then b else T	if a then b else T	if b then b else T
if T then b else F	if F then b else F	if a then b else F	if b then b else F
⋮	⋮	⋮	⋮

The haystack/$\simeq$

T, F, a, b,

T	T	T	T
T	F	a	b
T	a	a	$\texttt{if}\ a\ \texttt{then}\ T\ \texttt{else}\ b$
T	b	$\texttt{if}\ a\ \texttt{then}\ T\ \texttt{else}\ b$	b
F	T	$\texttt{if}\ a\ \texttt{then}\ F\ \texttt{else}\ T$	$\texttt{if}\ b\ \texttt{then}\ F\ \texttt{else}\ T$
F	F	F	F
F	a	F	$\texttt{if}\ b\ \texttt{then}\ F\ \texttt{else}\ a$
F	b	$\texttt{if}\ a\ \texttt{then}\ F\ \texttt{else}\ b$	F
a	T	T	$\texttt{if}\ b\ \texttt{then}\ a\ \texttt{else}\ T$
a	F	a	$\texttt{if}\ a\ \texttt{then}\ b\ \texttt{else}\ F$
a	a	a	a
a	b	$\texttt{if}\ a\ \texttt{then}\ T\ \texttt{else}\ b$	$\texttt{if}\ a\ \texttt{then}\ b\ \texttt{else}\ F$
b	T	$\texttt{if}\ a\ \texttt{then}\ b\ \texttt{else}\ T$	T
b	F	$\texttt{if}\ a\ \texttt{then}\ b\ \texttt{else}\ F$	b
$\vdots$	$\vdots$	$\vdots$	$\vdots$

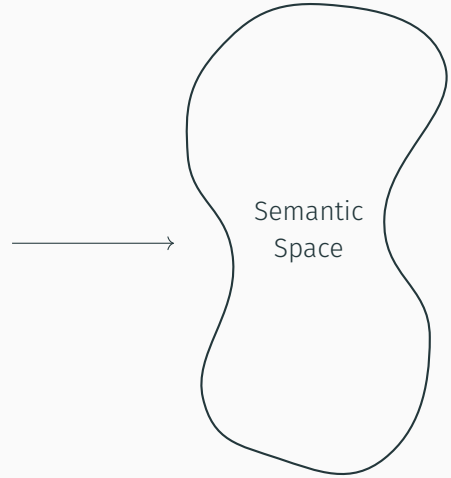
The haystack/$\simeq$

T, F, a, b,

T	T	T	T
T	F	a	b
T	a	a	`if` a `then` T `else` b
T	b	`if` a `then` T `else` b	b
F	T	`if` a `then` F `else` T	`if` b `then` F `else` T
F	F	F	F
F	a	F	`if` b `then` F `else` a
F	b	`if` a `then` F `else` b	F
a	T	T	`if` b `then` a `else` T
a	F	a	`if` a `then` b `else` F
a	a	a	a
a	b	`if` a `then` T `else` b	`if` a `then` b `else` F
b	T	`if` a `then` b `else` T	T
b	F	`if` a `then` b `else` F	b
⋮	⋮	⋮	⋮

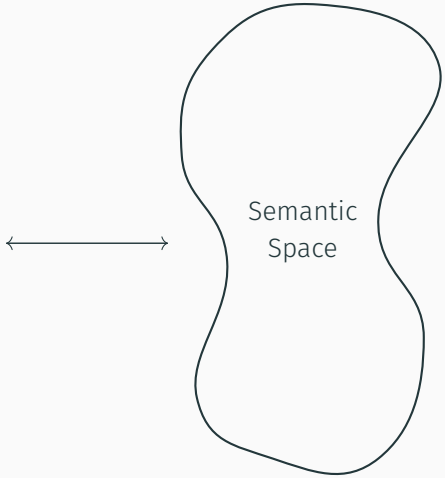
The haystack/$\simeq$

T
F
a
b
if a **then** T **else** b
if a **then** F **else** T
if a **then** F **else** b
if a **then** b **else** T
if a **then** b **else** F
if b **then** F **else** T
if b **then** F **else** a
if b **then** a **else** T
$\vdots$

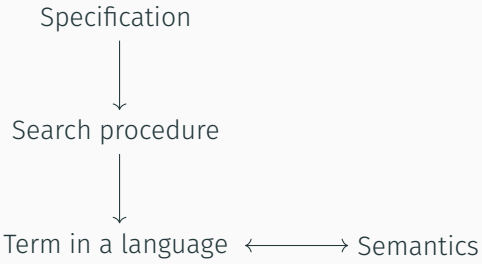


The haystack/$\simeq$

T
F
a
b
if a **then** T **else** b
if a **then** F **else** T
if a **then** F **else** b
if a **then** b **else** T
if a **then** b **else** F
if b **then** F **else** T
if b **then** F **else** a
if b **then** a **else** T
$\vdots$

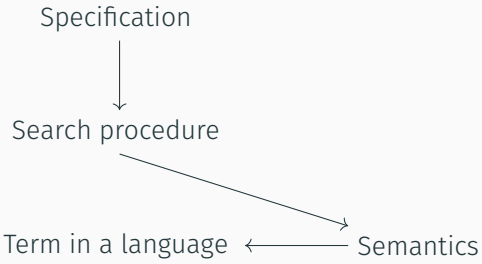


Program Synthesis by Semantics



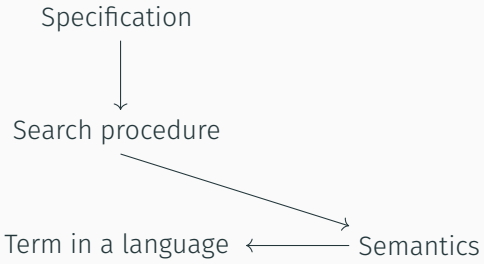
- **Soundness:** if the tool produces a program, the program is correct.
- **Completeness:** if a correct program could exist, the tool can produce it.

Program Synthesis by Semantics



- **Soundness:** if the tool produces a program, the program is correct.
- **Completeness:** if a correct program could exist, the tool can produce it.

Program Synthesis by Semantics



- **Soundness:** if the tool produces a program, the program is correct.
- **Completeness:** if a correct program could exist, the tool can produce it.
- **Semantic Optimality:** every term considered by the search procedure is semantically unique.

Category Theory Jumpscare

Denotations

Denotational semantics: mapping programs into a (mathematical) theory.

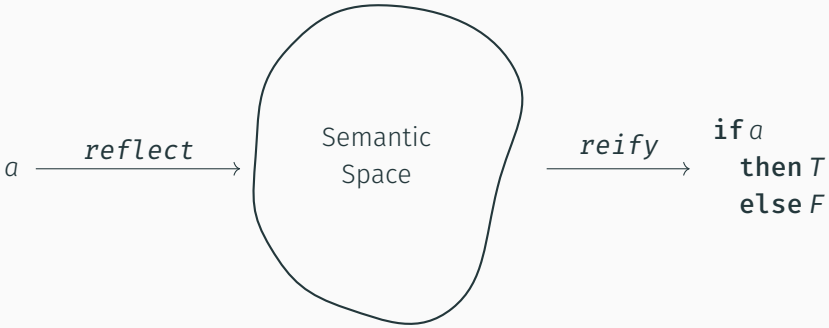
Categorical semantics: mapping programs into category theory.

- An interpretation function $[\![-]\!]$ takes "programming concepts" to categorical concepts,
- A type A becomes an object $[\![A]\!]$
- A context $\Gamma = x_1 : A_1, x_2 : A_2, \ldots$ becomes an object $[\![\Gamma]\!] = [\![A_1]\!] \times [\![A_2]\!] \times \ldots$
- A term $\Gamma \vdash t : A$ becomes a *morphism* $[\![t]\!] : [\![\Gamma]\!] \to [\![A]\!]$
- And the semantics of the language is reflected in the nature of the category:

$$t_1 \simeq t_2 \iff [\![t_1]\!] = [\![t_2]\!]$$

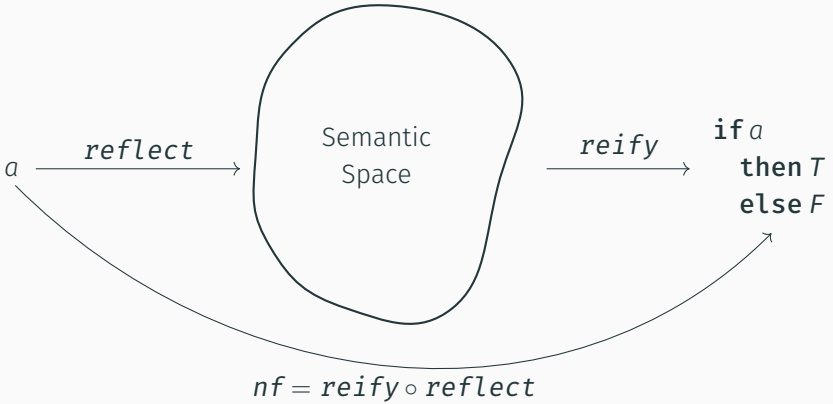
Normalisation by Evaluation

How do we extract (usable) semantics from the category? *Normal forms*

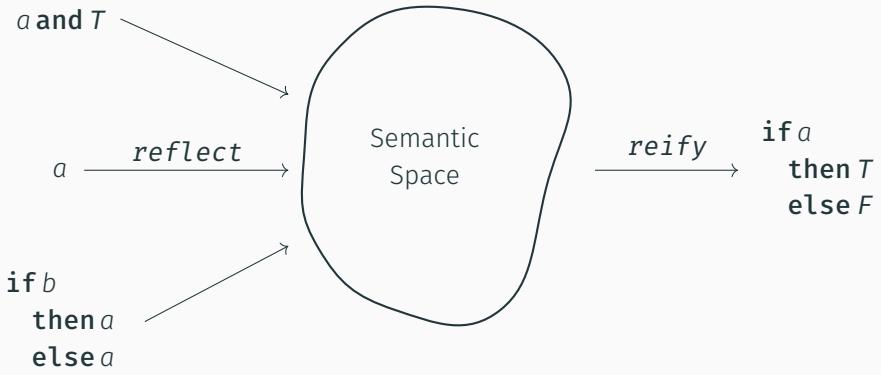


Normalisation by Evaluation

How do we extract (usable) semantics from the category? *Normal forms*



Normalisation by Evaluation



Normalisation Theorem: $t_1 \simeq t_2 \iff nf(t_1) = nf(t_2)$ ($\iff [\![t_1]\!] = [\![t_2]\!]$).

Normalisation by ~~Evaluation~~ Construction

The range of reify (and nf) are the *normal forms* of the language.

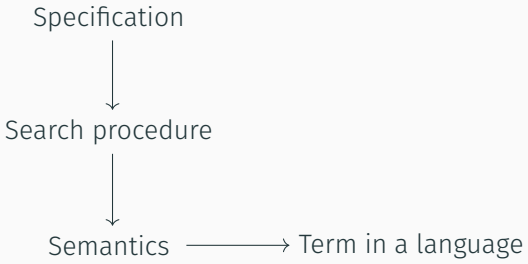
The Normalisation Theorem ($t_1 \simeq t_2 \iff \mathit{nf}(t_1) = \mathit{nf}(t_2)$) says that each program semantics has *exactly one* normal form.

So, the normal forms make up a *semantically optimal* language:

$$\frac{\Gamma \vdash c : \mathbb{B} \quad \Gamma \vdash t_1 : A \quad \Gamma \vdash t_2 : A}{\Gamma \vdash \mathbf{if}\ c\ \mathbf{then}\ t_1\ \mathbf{else}\ t_2 : A} \quad \mapsto \quad \frac{\Gamma \vdash_{\mathrm{NE}} c : \mathbb{B} \quad \Gamma \vdash_{\mathrm{NF}} t_1 : A \quad \Gamma \vdash_{\mathrm{NF}} t_2 : A \quad t_1 \not\simeq t_2 \quad \text{when } t_i = \mathbf{if}\ c'\ \ldots, c \sqsubset c'}{\Gamma \vdash_{\mathrm{NF}} \mathbf{if}\ c\ \mathbf{then}\ t_1\ \mathbf{else}\ t_2 : A}$$

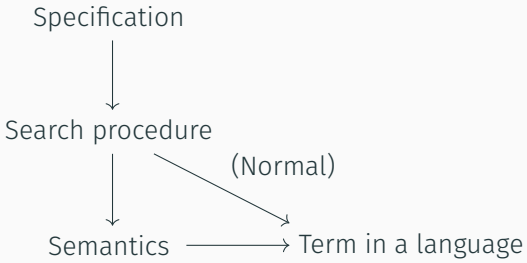
Normalised Synthesis

Program Synthesis by Semantics



- **Soundness:** if the tool produces a program, the program is correct.
- **Completeness:** if a correct program could exist, the tool can produce it.
- **Semantic Optimality:** every term considered by the search procedure is semantically unique.

Program Synthesis by Semantics



- **Soundness:** if the tool produces a program, the program is correct.
- **Completeness:** if a correct program could exist, the tool can produce it.
- **Semantic Optimality:** every term considered by the search procedure is semantically unique.

The normal-form haystack

T

F

$\texttt{if}\ a\ \texttt{then}\ T\ \texttt{else}\ F$

$\texttt{if}\ a\ \texttt{then}\ F\ \texttt{else}\ T$

$\texttt{if}\ b\ \texttt{then}\ T\ \texttt{else}\ F$

$\texttt{if}\ b\ \texttt{then}\ F\ \texttt{else}\ T$

$\texttt{if}\ a\ \texttt{then}\ (\texttt{if}\ b\ \texttt{then}\ T\ \texttt{else}\ F)\ \texttt{else}\ T$

$\texttt{if}\ a\ \texttt{then}\ (\texttt{if}\ b\ \texttt{then}\ T\ \texttt{else}\ F)\ \texttt{else}\ F$

$\texttt{if}\ a\ \texttt{then}\ (\texttt{if}\ b\ \texttt{then}\ F\ \texttt{else}\ T)\ \texttt{else}\ T$

$\texttt{if}\ a\ \texttt{then}\ (\texttt{if}\ b\ \texttt{then}\ F\ \texttt{else}\ T)\ \texttt{else}\ F$

$\texttt{if}\ a\ \texttt{then}\ T\ \texttt{else}\ (\texttt{if}\ b\ \texttt{then}\ T\ \texttt{else}\ F)$

$\texttt{if}\ a\ \texttt{then}\ T\ \texttt{else}\ (\texttt{if}\ b\ \texttt{then}\ F\ \texttt{else}\ T)$

$\texttt{if}\ a\ \texttt{then}\ F\ \texttt{else}\ (\texttt{if}\ b\ \texttt{then}\ T\ \texttt{else}\ F)$

$\texttt{if}\ a\ \texttt{then}\ F\ \texttt{else}\ (\texttt{if}\ b\ \texttt{then}\ F\ \texttt{else}\ T)$

$\texttt{if}\ a\ \texttt{then}\ (\texttt{if}\ b\ \texttt{then}\ T\ \texttt{else}\ F)\ \texttt{else}\ (\texttt{if}\ b\ \texttt{then}\ F\ \texttt{else}\ T)$

$\texttt{if}\ a\ \texttt{then}\ (\texttt{if}\ b\ \texttt{then}\ F\ \texttt{else}\ T)\ \texttt{else}\ (\texttt{if}\ b\ \texttt{then}\ T\ \texttt{else}\ F)$

The normal-form haystack

T

F

$\texttt{if}\ a\ \texttt{then}\ T\ \texttt{else}\ F$

$\texttt{if}\ a\ \texttt{then}\ F\ \texttt{else}\ T$

$\texttt{if}\ b\ \texttt{then}\ T\ \texttt{else}\ F$

$\texttt{if}\ b\ \texttt{then}\ F\ \texttt{else}\ T$

$\texttt{if}\ a\ \texttt{then}\ (\texttt{if}\ b\ \texttt{then}\ T\ \texttt{else}\ F)\ \texttt{else}\ T$

$\texttt{if}\ a\ \texttt{then}\ (\texttt{if}\ b\ \texttt{then}\ T\ \texttt{else}\ F)\ \texttt{else}\ F$

$\texttt{if}\ a\ \texttt{then}\ (\texttt{if}\ b\ \texttt{then}\ F\ \texttt{else}\ T)\ \texttt{else}\ T$

$\texttt{if}\ a\ \texttt{then}\ (\texttt{if}\ b\ \texttt{then}\ F\ \texttt{else}\ T)\ \texttt{else}\ F$

$\texttt{if}\ a\ \texttt{then}\ T\ \texttt{else}\ (\texttt{if}\ b\ \texttt{then}\ T\ \texttt{else}\ F)$

$\texttt{if}\ a\ \texttt{then}\ T\ \texttt{else}\ (\texttt{if}\ b\ \texttt{then}\ F\ \texttt{else}\ T)$

$\texttt{if}\ a\ \texttt{then}\ F\ \texttt{else}\ (\texttt{if}\ b\ \texttt{then}\ T\ \texttt{else}\ F)$

$\texttt{if}\ a\ \texttt{then}\ F\ \texttt{else}\ (\texttt{if}\ b\ \texttt{then}\ F\ \texttt{else}\ T)$

$\texttt{if}\ a\ \texttt{then}\ (\texttt{if}\ b\ \texttt{then}\ T\ \texttt{else}\ F)\ \texttt{else}\ (\texttt{if}\ b\ \texttt{then}\ F\ \texttt{else}\ T)$

$\texttt{if}\ a\ \texttt{then}\ (\texttt{if}\ b\ \texttt{then}\ F\ \texttt{else}\ T)\ \texttt{else}\ (\texttt{if}\ b\ \texttt{then}\ T\ \texttt{else}\ F)$

$\texttt{if}\ (\texttt{if}\ a\ \texttt{then}\ F\ \texttt{else}\ T)\ \texttt{then}\ b\ \texttt{else}\ F$

The normal-form haystack

T

F

$\texttt{if}\ a\ \texttt{then}\ T\ \texttt{else}\ F$

$\texttt{if}\ a\ \texttt{then}\ F\ \texttt{else}\ T$

$\texttt{if}\ b\ \texttt{then}\ T\ \texttt{else}\ F$

$\texttt{if}\ b\ \texttt{then}\ F\ \texttt{else}\ T$

$\texttt{if}\ a\ \texttt{then}\ (\texttt{if}\ b\ \texttt{then}\ T\ \texttt{else}\ F)\ \texttt{else}\ T$

$\texttt{if}\ a\ \texttt{then}\ (\texttt{if}\ b\ \texttt{then}\ T\ \texttt{else}\ F)\ \texttt{else}\ F$

$\texttt{if}\ a\ \texttt{then}\ (\texttt{if}\ b\ \texttt{then}\ F\ \texttt{else}\ T)\ \texttt{else}\ T$

$\texttt{if}\ a\ \texttt{then}\ (\texttt{if}\ b\ \texttt{then}\ F\ \texttt{else}\ T)\ \texttt{else}\ F$

$\texttt{if}\ a\ \texttt{then}\ T\ \texttt{else}\ (\texttt{if}\ b\ \texttt{then}\ T\ \texttt{else}\ F)$

$\texttt{if}\ a\ \texttt{then}\ T\ \texttt{else}\ (\texttt{if}\ b\ \texttt{then}\ F\ \texttt{else}\ T)$

$\texttt{if}\ a\ \texttt{then}\ F\ \texttt{else}\ (\texttt{if}\ b\ \texttt{then}\ T\ \texttt{else}\ F)$

$\texttt{if}\ a\ \texttt{then}\ F\ \texttt{else}\ (\texttt{if}\ b\ \texttt{then}\ F\ \texttt{else}\ T)$

$\texttt{if}\ a\ \texttt{then}\ (\texttt{if}\ b\ \texttt{then}\ T\ \texttt{else}\ F)\ \texttt{else}\ (\texttt{if}\ b\ \texttt{then}\ F\ \texttt{else}\ T)$

$\texttt{if}\ a\ \texttt{then}\ (\texttt{if}\ b\ \texttt{then}\ F\ \texttt{else}\ T)\ \texttt{else}\ (\texttt{if}\ b\ \texttt{then}\ T\ \texttt{else}\ F)$

$\texttt{if}\ (\texttt{if}\ a\ \texttt{then}\ F\ \texttt{else}\ T)\ \texttt{then}\ b\ \texttt{else}\ F$ 

The normal-form haystack

T

F

$\texttt{if}\ a\ \texttt{then}\ T\ \texttt{else}\ F$

$\texttt{if}\ a\ \texttt{then}\ F\ \texttt{else}\ T$

$\texttt{if}\ b\ \texttt{then}\ T\ \texttt{else}\ F$

$\texttt{if}\ b\ \texttt{then}\ F\ \texttt{else}\ T$

$\texttt{if}\ a\ \texttt{then}\ (\texttt{if}\ b\ \texttt{then}\ T\ \texttt{else}\ F)\ \texttt{else}\ T$

$\texttt{if}\ a\ \texttt{then}\ (\texttt{if}\ b\ \texttt{then}\ T\ \texttt{else}\ F)\ \texttt{else}\ F$

$\texttt{if}\ a\ \texttt{then}\ (\texttt{if}\ b\ \texttt{then}\ F\ \texttt{else}\ T)\ \texttt{else}\ T$

$\texttt{if}\ a\ \texttt{then}\ (\texttt{if}\ b\ \texttt{then}\ F\ \texttt{else}\ T)\ \texttt{else}\ F$

$\texttt{if}\ a\ \texttt{then}\ T\ \texttt{else}\ (\texttt{if}\ b\ \texttt{then}\ T\ \texttt{else}\ F)$

$\texttt{if}\ a\ \texttt{then}\ T\ \texttt{else}\ (\texttt{if}\ b\ \texttt{then}\ F\ \texttt{else}\ T)$

$\texttt{if}\ a\ \texttt{then}\ F\ \texttt{else}\ (\texttt{if}\ b\ \texttt{then}\ T\ \texttt{else}\ F)$

$\texttt{if}\ a\ \texttt{then}\ F\ \texttt{else}\ (\texttt{if}\ b\ \texttt{then}\ F\ \texttt{else}\ T)$

$\texttt{if}\ a\ \texttt{then}\ (\texttt{if}\ b\ \texttt{then}\ T\ \texttt{else}\ F)\ \texttt{else}\ (\texttt{if}\ b\ \texttt{then}\ F\ \texttt{else}\ T)$

$\texttt{if}\ a\ \texttt{then}\ (\texttt{if}\ b\ \texttt{then}\ F\ \texttt{else}\ T)\ \texttt{else}\ (\texttt{if}\ b\ \texttt{then}\ T\ \texttt{else}\ F)$

$\texttt{if}\ (\texttt{if}\ a\ \texttt{then}\ F\ \texttt{else}\ T)\ \texttt{then}\ b\ \texttt{else}\ F$ ✗

$\texttt{if}\ b\ \texttt{then}\ (\texttt{if}\ a\ \texttt{then}\ T\ \texttt{else}\ F)\ \texttt{else}\ T$

The normal-form haystack

T

F

$\texttt{if}\ a\ \texttt{then}\ T\ \texttt{else}\ F$

$\texttt{if}\ a\ \texttt{then}\ F\ \texttt{else}\ T$

$\texttt{if}\ b\ \texttt{then}\ T\ \texttt{else}\ F$

$\texttt{if}\ b\ \texttt{then}\ F\ \texttt{else}\ T$

$\texttt{if}\ a\ \texttt{then}\ (\texttt{if}\ b\ \texttt{then}\ T\ \texttt{else}\ F)\ \texttt{else}\ T$

$\texttt{if}\ a\ \texttt{then}\ (\texttt{if}\ b\ \texttt{then}\ T\ \texttt{else}\ F)\ \texttt{else}\ F$

$\texttt{if}\ a\ \texttt{then}\ (\texttt{if}\ b\ \texttt{then}\ F\ \texttt{else}\ T)\ \texttt{else}\ T$

$\texttt{if}\ a\ \texttt{then}\ (\texttt{if}\ b\ \texttt{then}\ F\ \texttt{else}\ T)\ \texttt{else}\ F$

$\texttt{if}\ a\ \texttt{then}\ T\ \texttt{else}\ (\texttt{if}\ b\ \texttt{then}\ T\ \texttt{else}\ F)$

$\texttt{if}\ a\ \texttt{then}\ T\ \texttt{else}\ (\texttt{if}\ b\ \texttt{then}\ F\ \texttt{else}\ T)$

$\texttt{if}\ a\ \texttt{then}\ F\ \texttt{else}\ (\texttt{if}\ b\ \texttt{then}\ T\ \texttt{else}\ F)$

$\texttt{if}\ a\ \texttt{then}\ F\ \texttt{else}\ (\texttt{if}\ b\ \texttt{then}\ F\ \texttt{else}\ T)$

$\texttt{if}\ a\ \texttt{then}\ (\texttt{if}\ b\ \texttt{then}\ T\ \texttt{else}\ F)\ \texttt{else}\ (\texttt{if}\ b\ \texttt{then}\ F\ \texttt{else}\ T)$

$\texttt{if}\ a\ \texttt{then}\ (\texttt{if}\ b\ \texttt{then}\ F\ \texttt{else}\ T)\ \texttt{else}\ (\texttt{if}\ b\ \texttt{then}\ T\ \texttt{else}\ F)$



The normal-form haystack

T

F

$\texttt{if}\ a\ \texttt{then}\ T\ \texttt{else}\ F$

$\texttt{if}\ a\ \texttt{then}\ F\ \texttt{else}\ T$

$\texttt{if}\ b\ \texttt{then}\ T\ \texttt{else}\ F$

$\texttt{if}\ b\ \texttt{then}\ F\ \texttt{else}\ T$

$\texttt{if}\ a\ \texttt{then}\ (\texttt{if}\ b\ \texttt{then}\ T\ \texttt{else}\ F)\ \texttt{else}\ T$

$\texttt{if}\ a\ \texttt{then}\ (\texttt{if}\ b\ \texttt{then}\ T\ \texttt{else}\ F)\ \texttt{else}\ F$

$\texttt{if}\ a\ \texttt{then}\ (\texttt{if}\ b\ \texttt{then}\ F\ \texttt{else}\ T)\ \texttt{else}\ T$

$\texttt{if}\ a\ \texttt{then}\ (\texttt{if}\ b\ \texttt{then}\ F\ \texttt{else}\ T)\ \texttt{else}\ F$

$\texttt{if}\ a\ \texttt{then}\ T\ \texttt{else}\ (\texttt{if}\ b\ \texttt{then}\ T\ \texttt{else}\ F)$

$\texttt{if}\ a\ \texttt{then}\ T\ \texttt{else}\ (\texttt{if}\ b\ \texttt{then}\ F\ \texttt{else}\ T)$

$\texttt{if}\ a\ \texttt{then}\ F\ \texttt{else}\ (\texttt{if}\ b\ \texttt{then}\ T\ \texttt{else}\ F)$

$\texttt{if}\ a\ \texttt{then}\ F\ \texttt{else}\ (\texttt{if}\ b\ \texttt{then}\ F\ \texttt{else}\ T)$

$\texttt{if}\ a\ \texttt{then}\ (\texttt{if}\ b\ \texttt{then}\ T\ \texttt{else}\ F)\ \texttt{else}\ (\texttt{if}\ b\ \texttt{then}\ F\ \texttt{else}\ T)$

$\texttt{if}\ a\ \texttt{then}\ (\texttt{if}\ b\ \texttt{then}\ F\ \texttt{else}\ T)\ \texttt{else}\ (\texttt{if}\ b\ \texttt{then}\ T\ \texttt{else}\ F)$

~~$\texttt{if}\ (\texttt{if}\ a\ \texttt{then}\ F\ \texttt{else}\ T)\ \texttt{then}\ b\ \texttt{else}\ F$~~

~~$\texttt{if}\ b\ \texttt{then}\ (\texttt{if}\ a\ \texttt{then}\ T\ \texttt{else}\ F)\ \texttt{else}\ T$~~

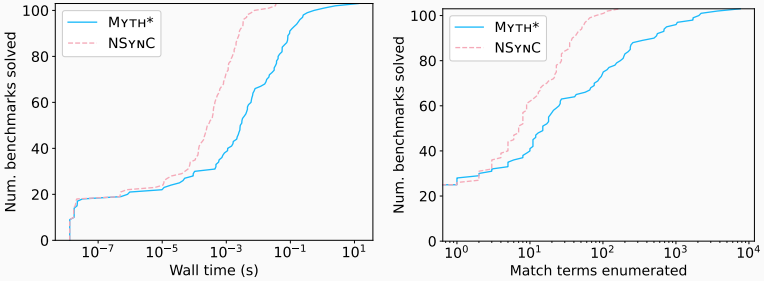
$\texttt{if}\ a\ \texttt{then}\ F\ \texttt{else}\ F$

The normal-form haystack

```
T
F
if a then T else F
if a then F else T
if b then T else F
if b then F else T
if a then (if b then T else F) else T
if a then (if b then T else F) else F
if a then (if b then F else T) else T
if a then (if b then F else T) else F
if a then T else (if b then T else F)
if a then T else (if b then F else T)
if a then F else (if b then T else F)
if a then F else (if b then F else T)
if a then (if b then T else F) else (if b then F else T)
if a then (if b then F else T) else (if b then T else F)
```

```
if (if a then F else T) then b else F
if b then (if a then T else F) else T
        if a then F else F
```

Normalised Synthesis



- **Soundness:** if NSynC produces a program, the program is correct.
- **Completeness:** if Myth can produce a correct program, so can NSynC.
- **Semantic Optimality:** every term considered by the search procedure is semantically unique.

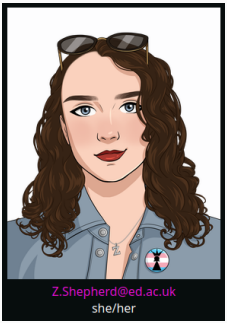
Conclusion

Summary

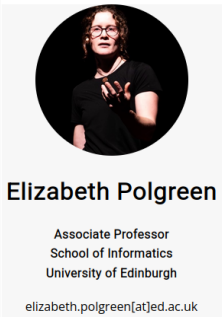
- **Synthesis:** Specifications and searching for programs
- **Semantics:** Different meanings of programs, and NbE
- **Normalised synthesis:** Synthesising in normal form to find programs more quickly

Plugs

Me:



My supervisor:



The PhD fair:

