Automating Algorithm Design through Genetic Programming Hyper-Heuristics

Elsa Browning

Division of Science and Mathematics
University of Minnesota, Morris
Morris, Minnesota, USA

April 15, 2017
Morris, MN

Overview
●○

Background

Hyper-heuristics

GP Variants
○○○○○○○○○○

Autoconstruction

Summary

What does the title mean?

- Reducing the human component in algorithm design

https://scratch.mit.edu/
discuss/m/topic/200574/

What does the title mean?

- Reducing the human component in algorithm design

- More work at the beginning, more possibilities

https://scratch.mit.edu/
discuss/m/topic/200574/

What does the title mean?

- Reducing the human component in algorithm design

- More work at the beginning, more possibilities

- Genetic programming hyper-heuristics as a method to the madness

https://scratch.mit.edu/
discuss/m/topic/200574/

Outline

1 **Background**

2 **Hyper-heuristics**

3 **Genetic Programming Variants**

4 **Autoconstruction**

5 **Summary**

Outline

Overview
○○

Background

Hyper-heuristics

GP Variants
○○○○○○○○○○

Autoconstruction

Summary

Outline

Outline

Overview
○○
Background
Hyper-heuristics
GP Variants
●○○○○○○○○○○
Autoconstruction
Summary

Stack-based genetic programming

Data-stacks are used for managing
input and output of operations.

Programs are represented as linear
sequences of literals and instructions.
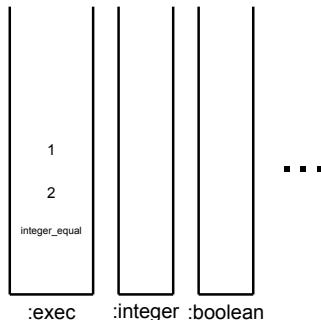Below is an example of a simple Push
program:

```
(1 2 integer_equal)
```


:exec   :integer :boolean

Overview
○○
Background
Hyper-heuristics
GP Variants
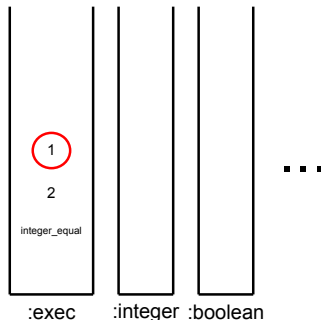○●○○○○○○○○○
Autoconstruction
Summary

Stack-based genetic programming

Data-stacks are used for managing
input and output of operations.

Programs are represented as linear
sequences of literals and instructions.
Below is an example of a simple Push
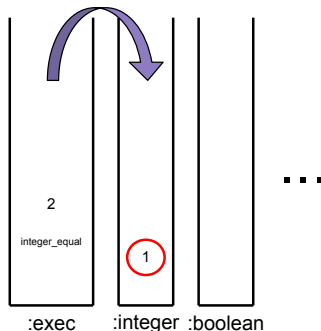program:

```
(1 2 integer_equal)
```

Stack-based genetic programming

Data-stacks are used for managing
input and output of operations.

Programs are represented as linear
sequences of literals and instructions.
Below is an example of a simple Push
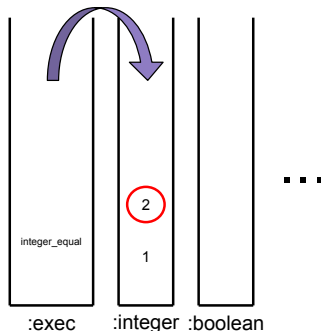program:

```
(1 2 integer_equal)
```

Stack-based genetic programming

Data-stacks are used for managing
input and output of operations.

Programs are represented as linear
sequences of literals and instructions.
Below is an example of a simple Push
program:

```
(1 2 integer_equal)
```

Stack-based genetic programming

Data-stacks are used for managing
input and output of operations.

Programs are represented as linear
sequences of literals and instructions.
Below is an example of a simple Push
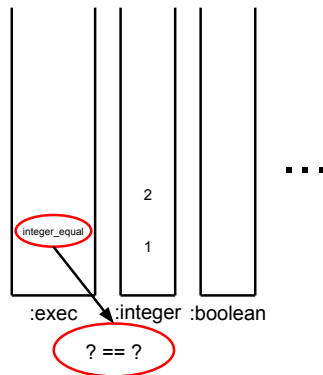program:

```
(1 2 integer_equal)
```

Stack-based genetic programming

Data-stacks are used for managing
input and output of operations.

Programs are represented as linear
sequences of literals and instructions.
Below is an example of a simple Push
program:

```
(1 2 integer_equal)
```

Stack-based genetic programming

Data-stacks are used for managing input and output of operations.

Programs are represented as linear sequences of literals and instructions. Below is an example of a simple Push program:
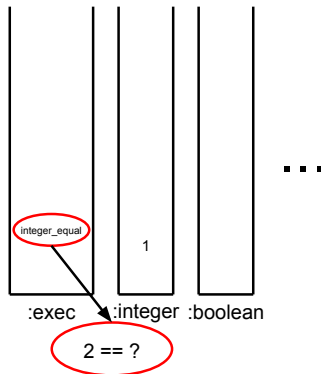
```
(1 2 integer_equal)
```

Stack-based genetic programming

Data-stacks are used for managing
input and output of operations.

Programs are represented as linear
sequences of literals and instructions.
Below is an example of a simple Push
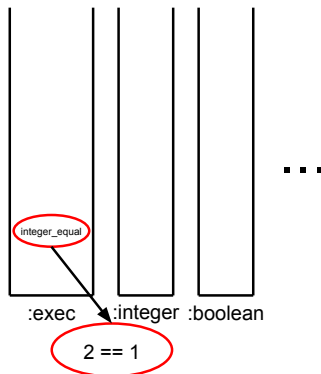program:

```
(1 2 integer_equal)
```

Stack-based genetic programming

Data-stacks are used for managing
input and output of operations.

Programs are represented as linear
sequences of literals and instructions.
Below is an example of a simple Push
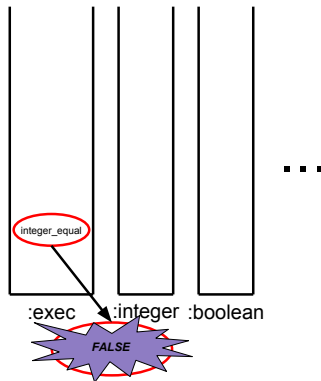program:

> (1 2 integer_equal)

Stack-based genetic programming

Data-stacks are used for managing
input and output of operations.

Programs are represented as linear
sequences of literals and instructions.
Below is an example of a simple Push
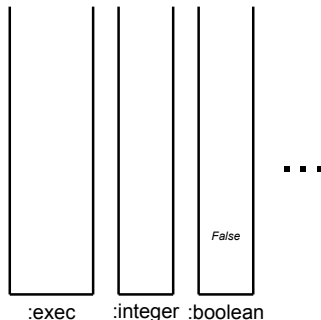program:

```
(1 2 integer_equal)
```

Stack-based genetic programming

Data-stacks are used for managing
input and output of operations.

Programs are represented as linear
sequences of literals and instructions.
Below is an example of a simple Push
program:

```
(1 2 integer_equal)
```


:exec :integer :boolean

Outline

Overview
○○

Background

Hyper-heuristics

GP Variants
○○○○○○○○○○○

Autoconstruction

Summary

Outline

1 **Background**

2 **Hyper-heuristics**

3 **Genetic Programming Variants**

4 **Autoconstruction**

5 **Summary**