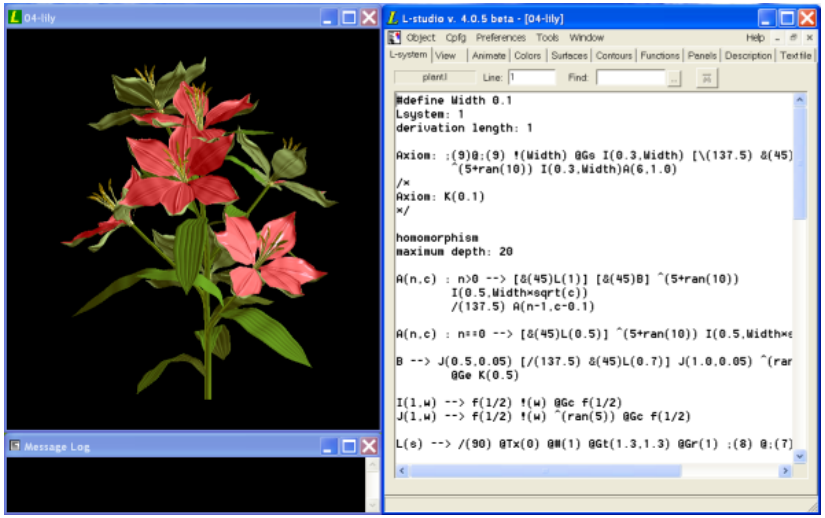
synthesizing programs that procedurally generate plant graphics

Caleb Winston
Mentored by Professor Rastislav Bodik
Programming Languages & Software Engineering (PLSE) Group

writing programs that generate lots of plant graphics is hard

Generated plants in virtual reality video game, No Man's Sky

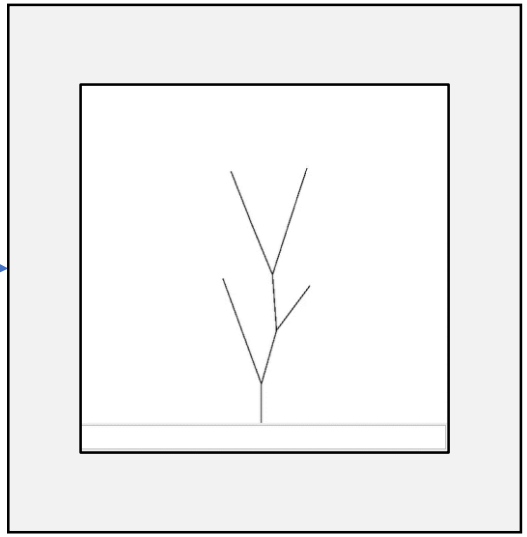


A program that can generate field of plants (right), one example of a generated plant that would belong to the field (left)

writing programs that generate lots of plant graphics is hard

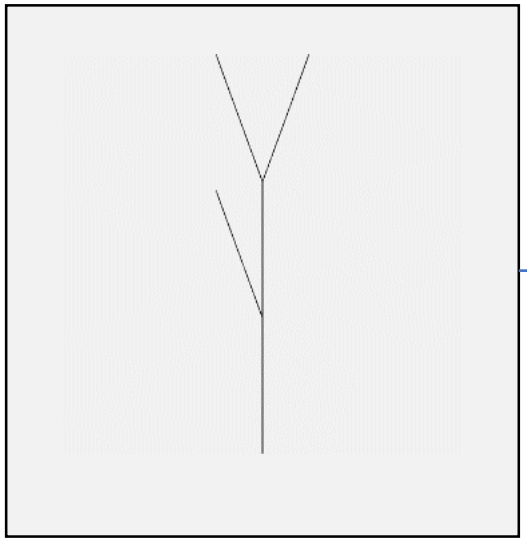
```
(rule F FF)
(rule X
F[[X]+X]+F
[+FX]-X]-X)
```

a program that generates graphics (an L-system)



graphics generated by program

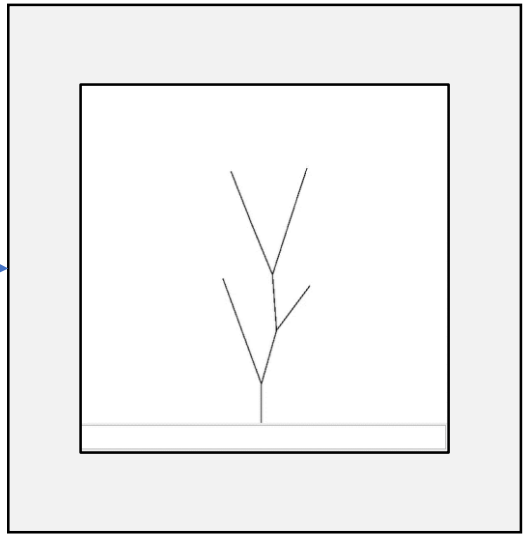
idea: make computer synthesize those programs from an example graphic provided by designer



an example of a graphic that a designer wants the program to generate

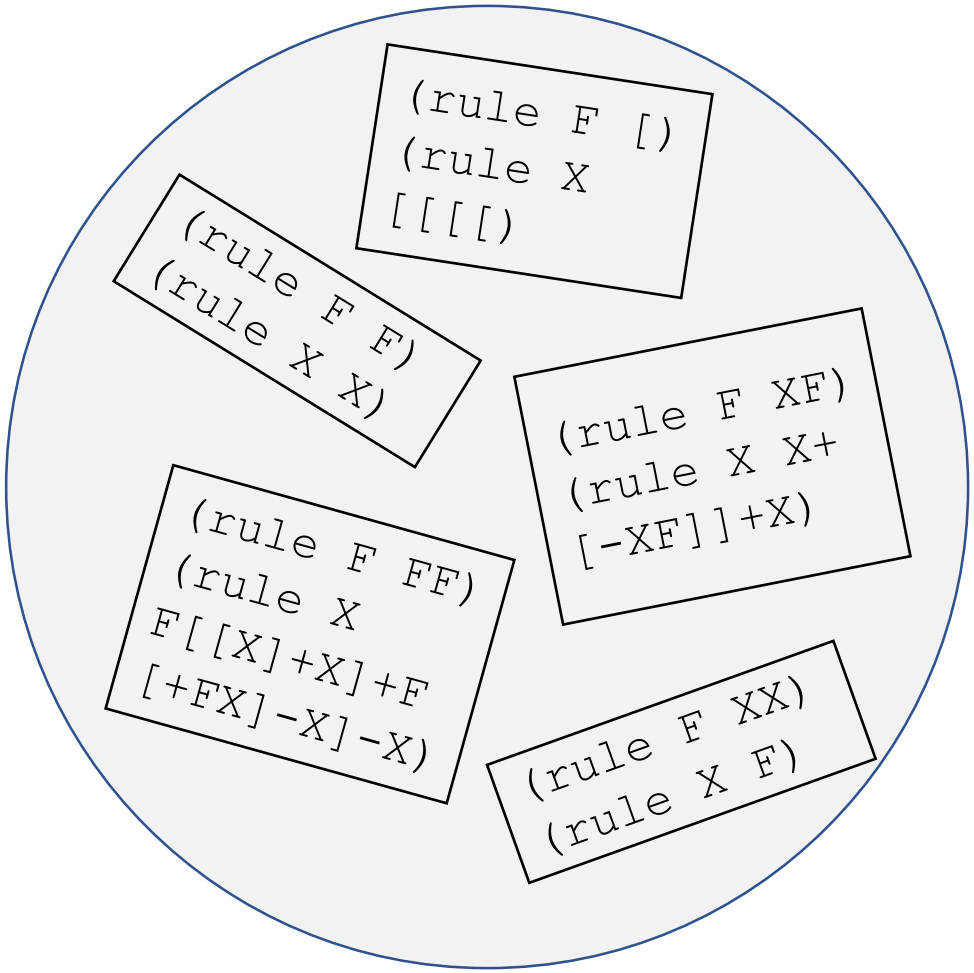
```
(rule F FF)
(rule X
F[[X]+X]+F
[+FX]-X]-X)
```

a program that generates graphics (an L-system)



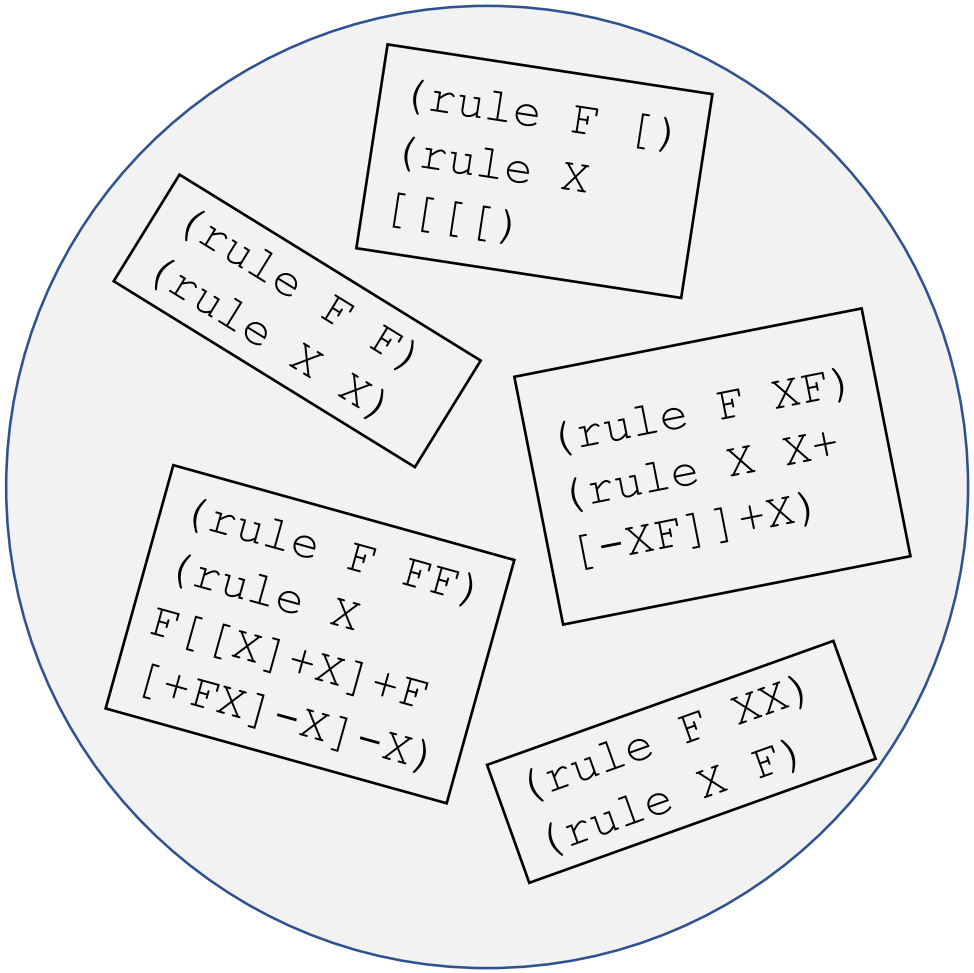
graphics generated by program

step 1: prune set of possible programs



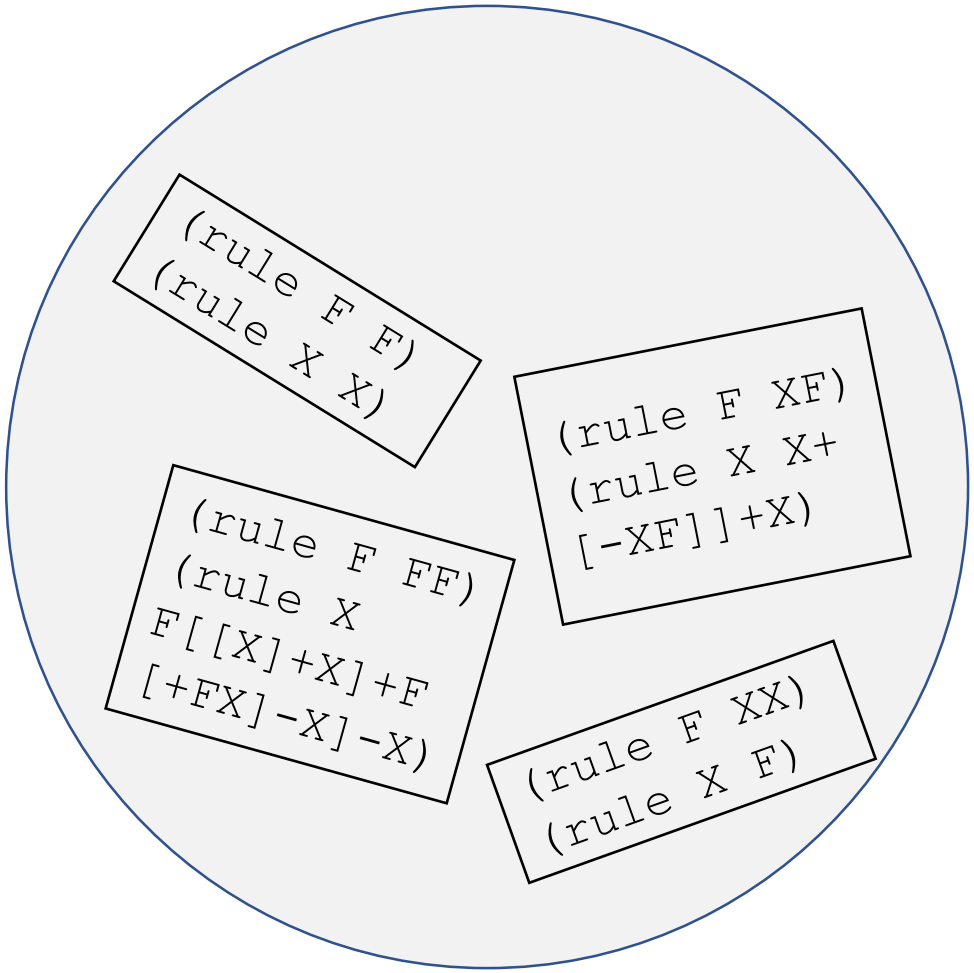
(0) take infinite set of all programs

step 1: prune set of possible programs



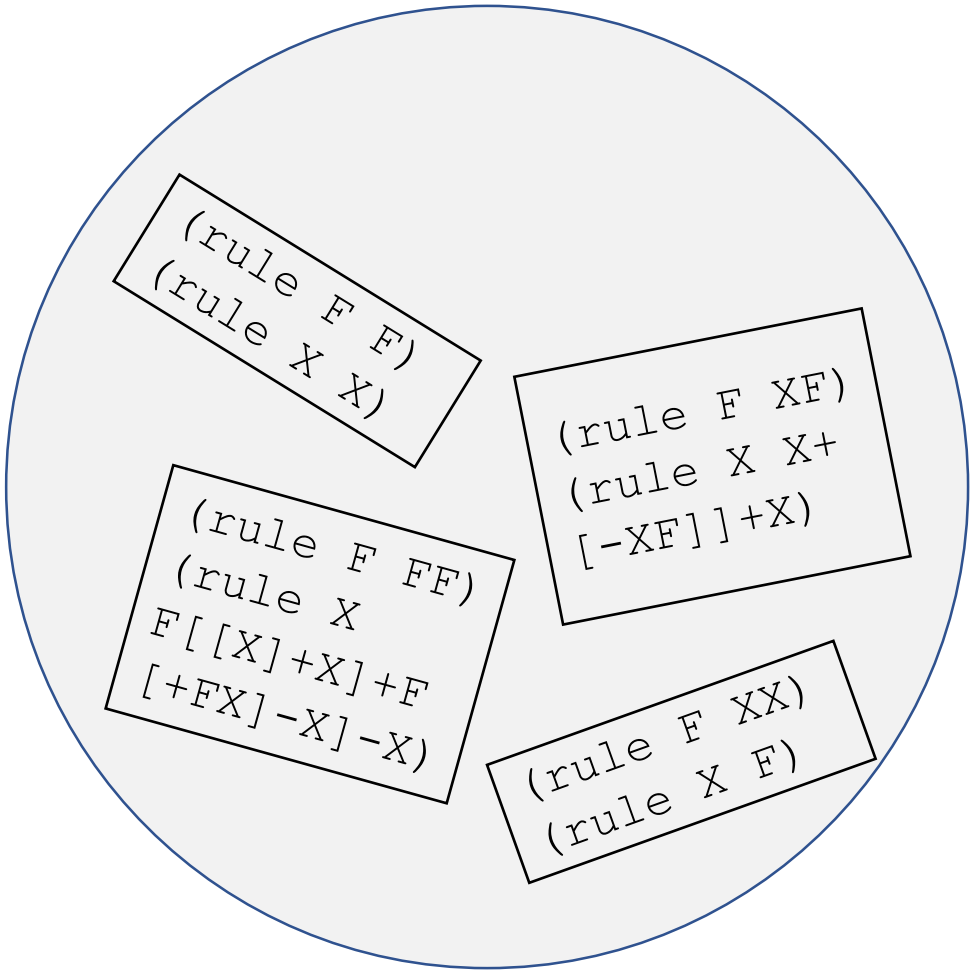
(1) remove programs that generate invalid plant parts

step 1: prune set of possible programs



(1) remove programs that generate invalid plant parts

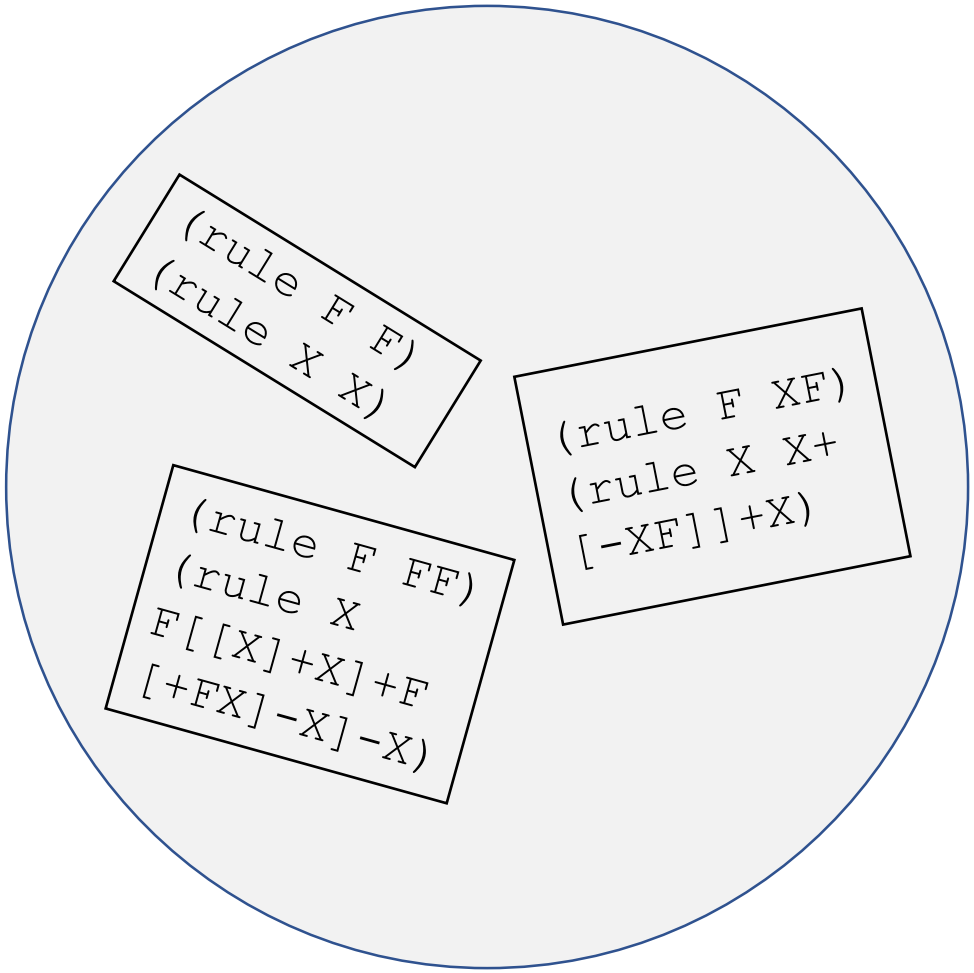
step 1: prune set of possible programs



(1) remove programs that generate invalid plant parts

(2) remove programs that generate plant parts that don't come from plant given by designer

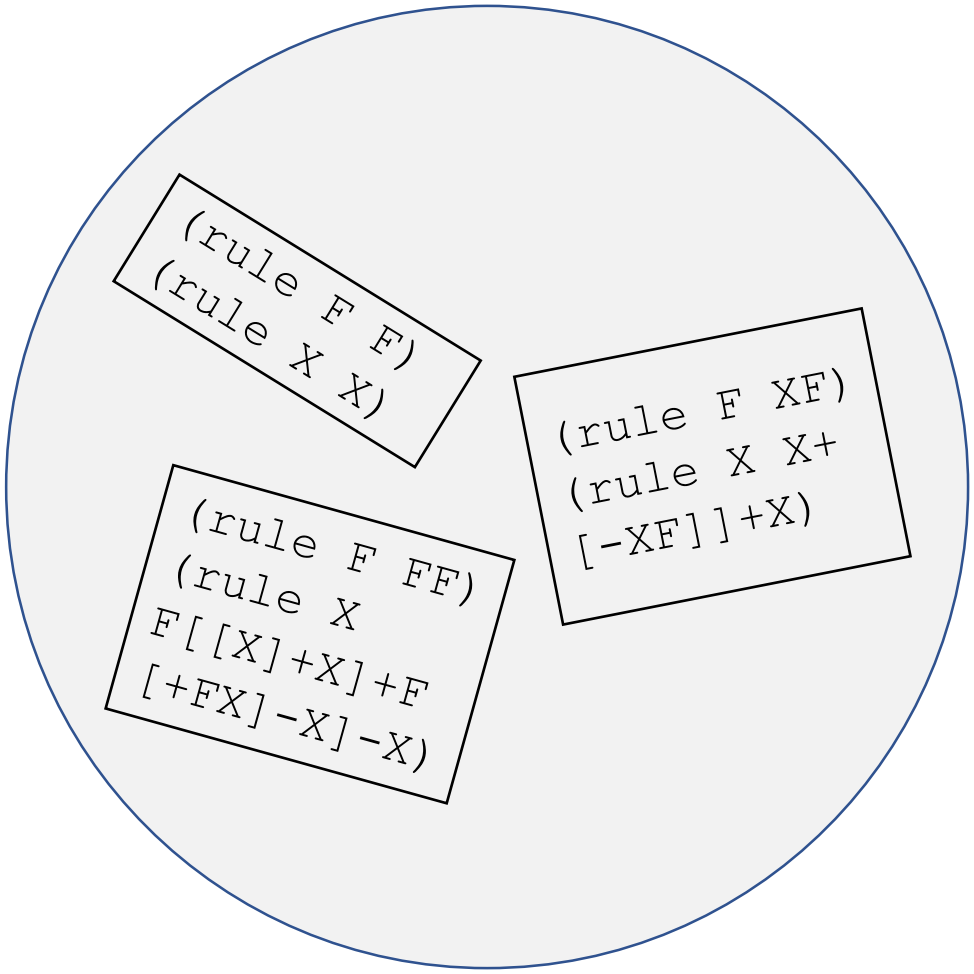
step 1: prune set of possible programs



(1) remove programs that generate invalid plant parts

(2) remove programs that generate plant parts that don't come from plant given by designer

step 1: prune set of possible programs

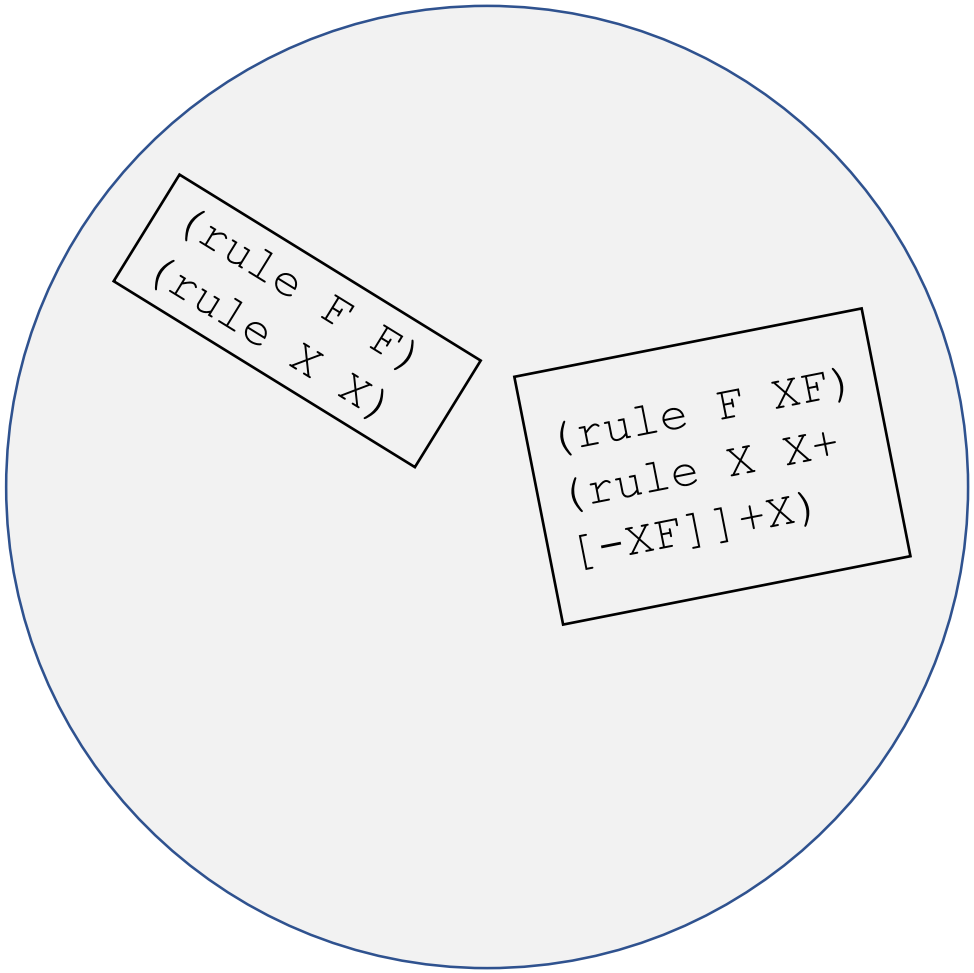


(1) remove programs that generate invalid plant parts

(2) remove programs that generate plant parts that don't come from plant given by designer

(3) Remove programs that generate plant parts that are too big

step 1: prune set of possible programs

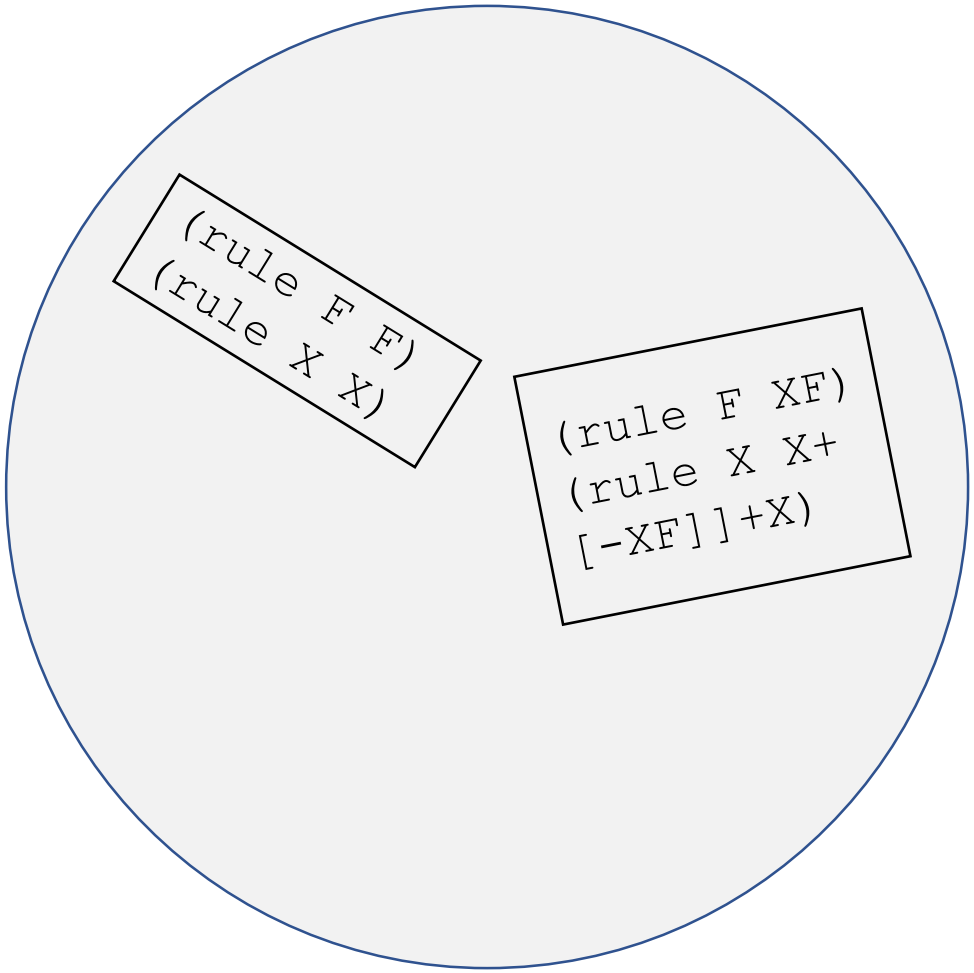


(1) remove programs that generate invalid plant parts

(2) remove programs that generate plant parts that don't come from plant given by designer

(3) Remove programs that generate plant parts that are too big

step 1: prune set of possible programs



(1) remove programs that generate invalid plant parts

(2) remove programs that generate plant parts that don't come from plant given by designer

(3) Remove programs that generate plant parts that are too big

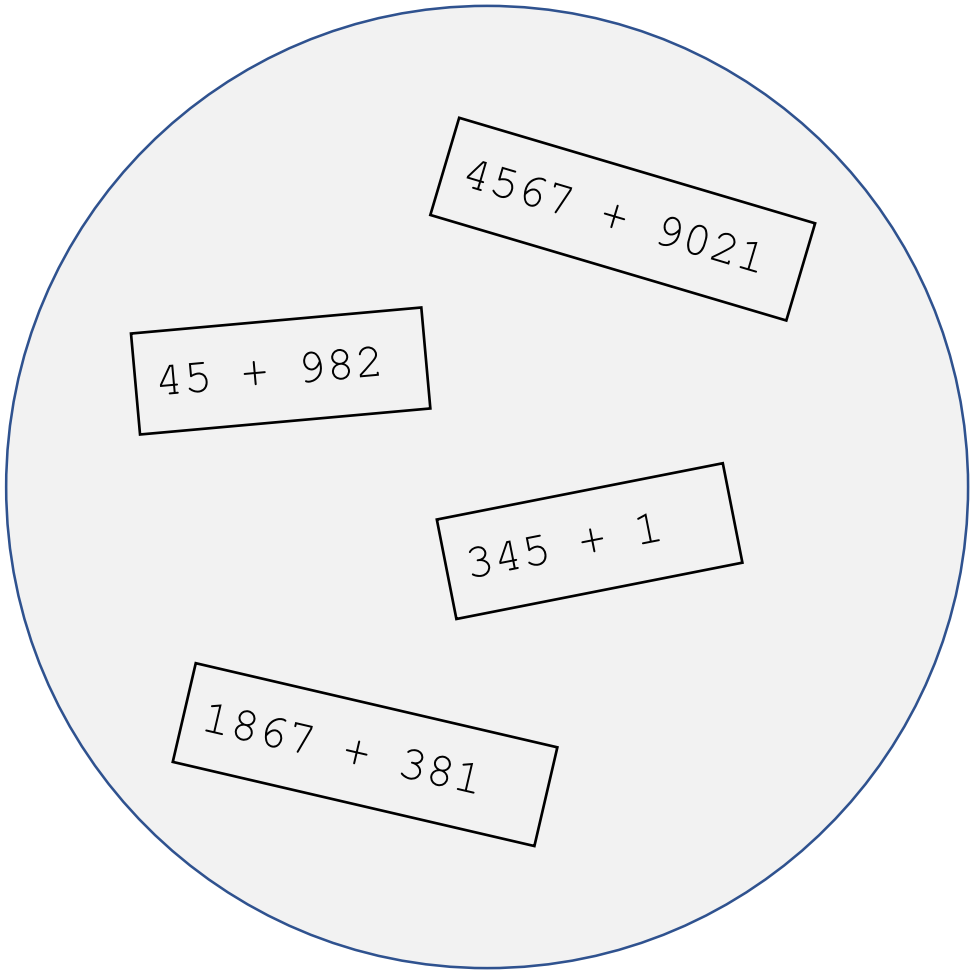
(4) smaller subset of possible programs

step 2: tell Rosette to constrain program

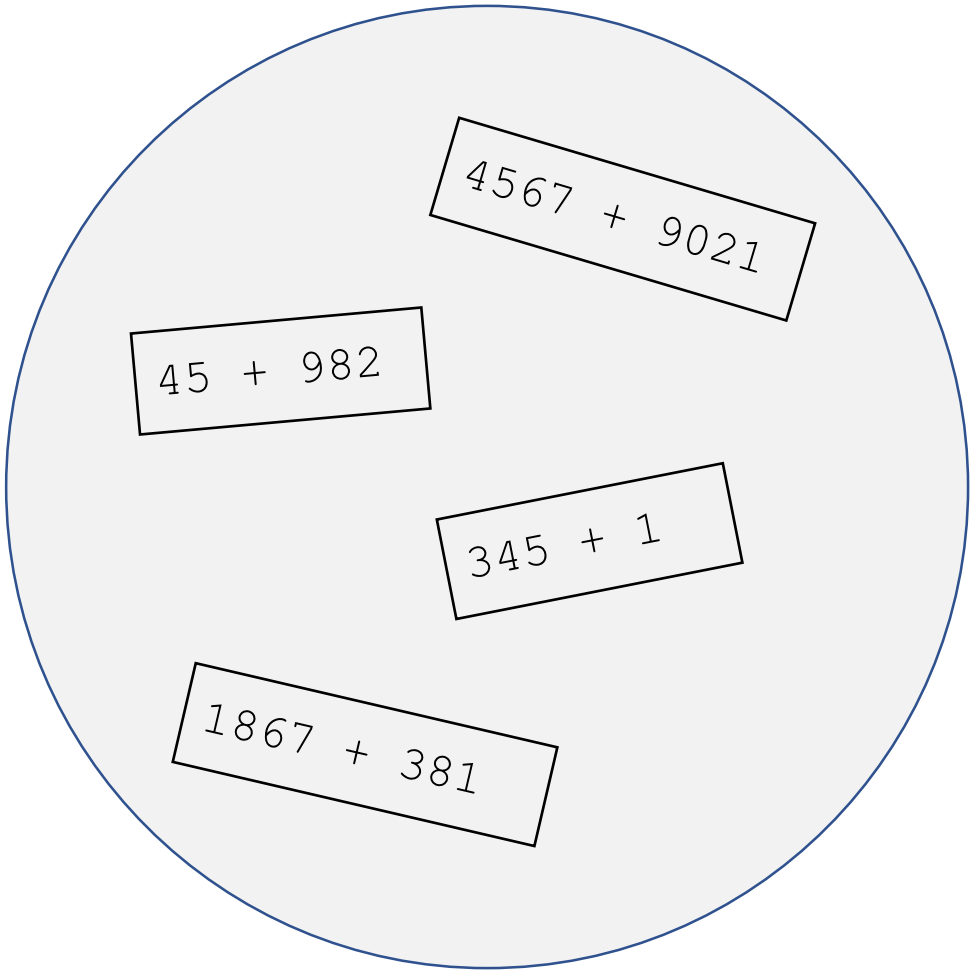
(Rosette is a framework for doing program synthesis, developed by Emina Torlak and Rastislav Bodik)

- define symbolic expression for a program using pruned set of possible programs

but what is a symbolic expression?

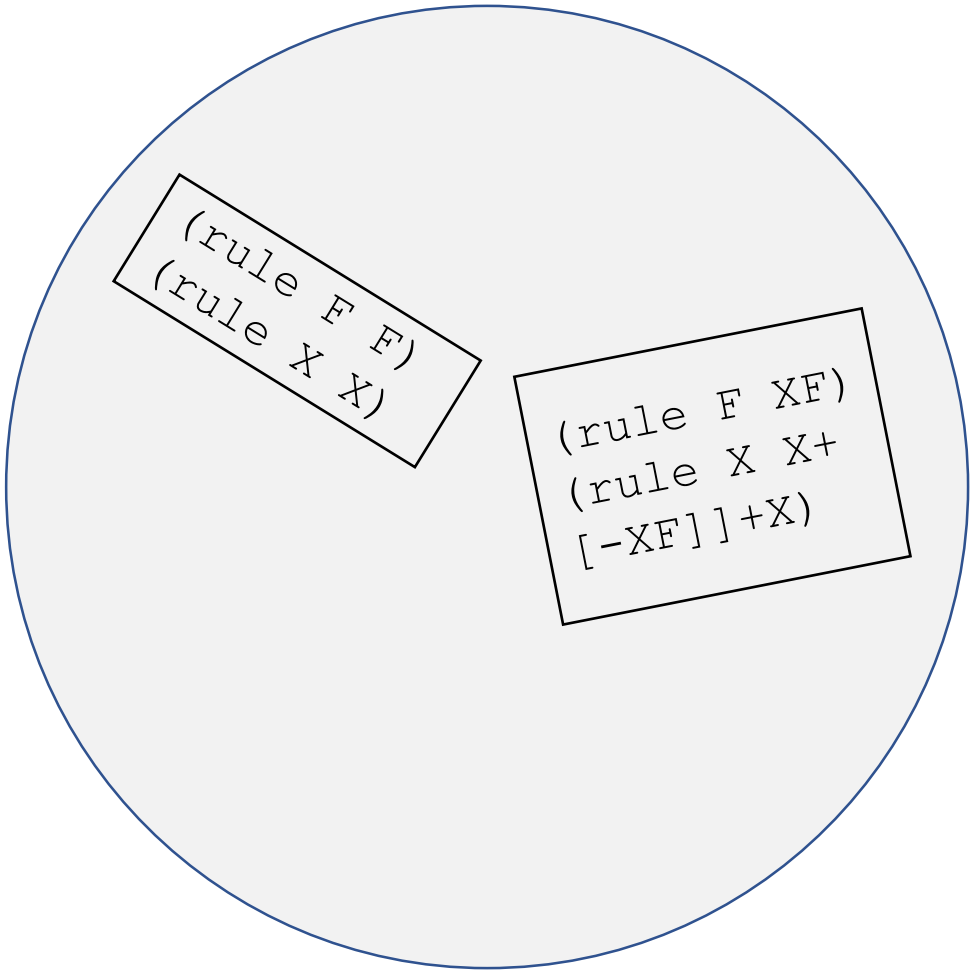


but what is a symbolic expression?

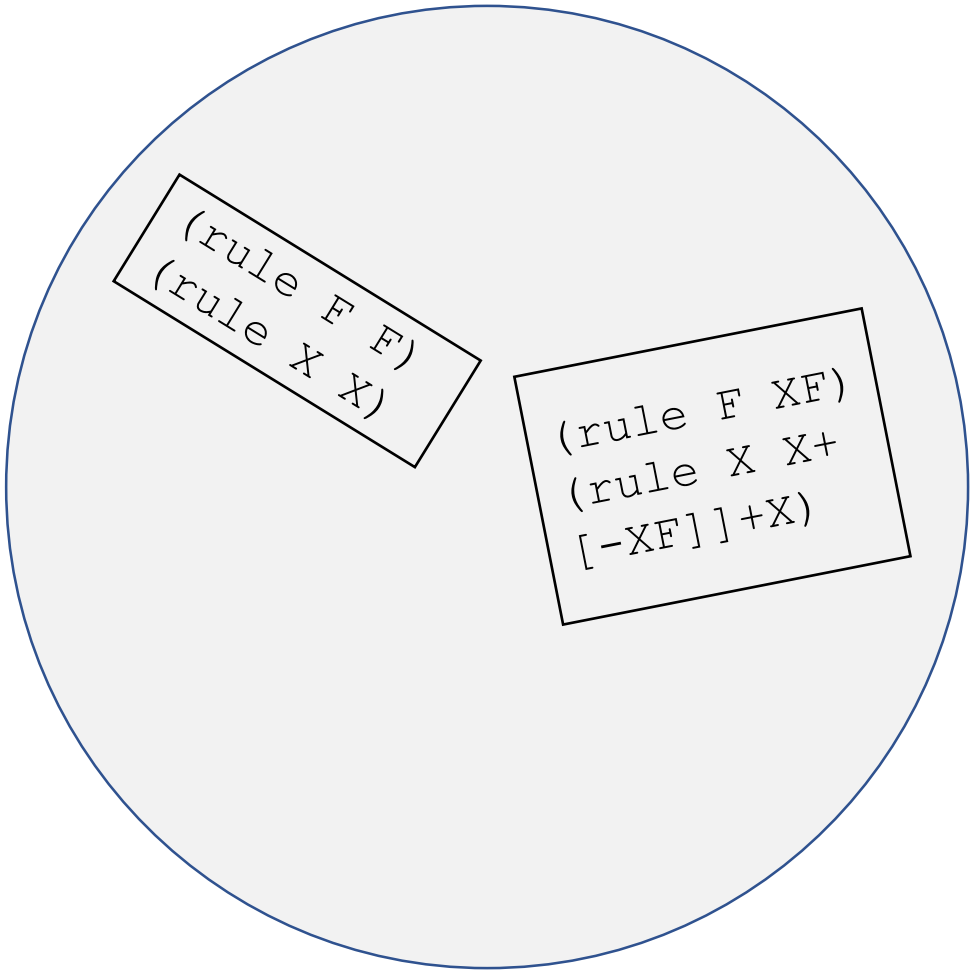


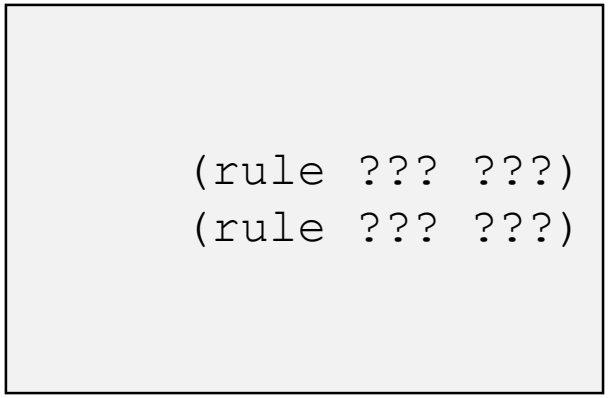
X + Y

but what is a symbolic expression?



but what is a symbolic expression?





step 2: tell Rosette to constrain program

(Rosette is a framework for doing program synthesis, developed by Emina Torlak and Rastislav Bodik)

- define symbolic expression for a program

step 2: tell Rosette to constrain program

(Rosette is a framework for doing program synthesis, developed by Emina Torlak and Rastislav Bodik)

- define symbolic expression for a program
- define symbolic expression for output plant graphics from program

step 2: tell Rosette to constrain program

(Rosette is a framework for doing program synthesis, developed by Emina Torlak and Rastislav Bodik)

- define symbolic expression for a program
- define symbolic expression for output plant graphics from program
- define symbolic expression for # of edits between output plant graphic and plant graphic given by designer

step 2: tell Rosette to constrain program

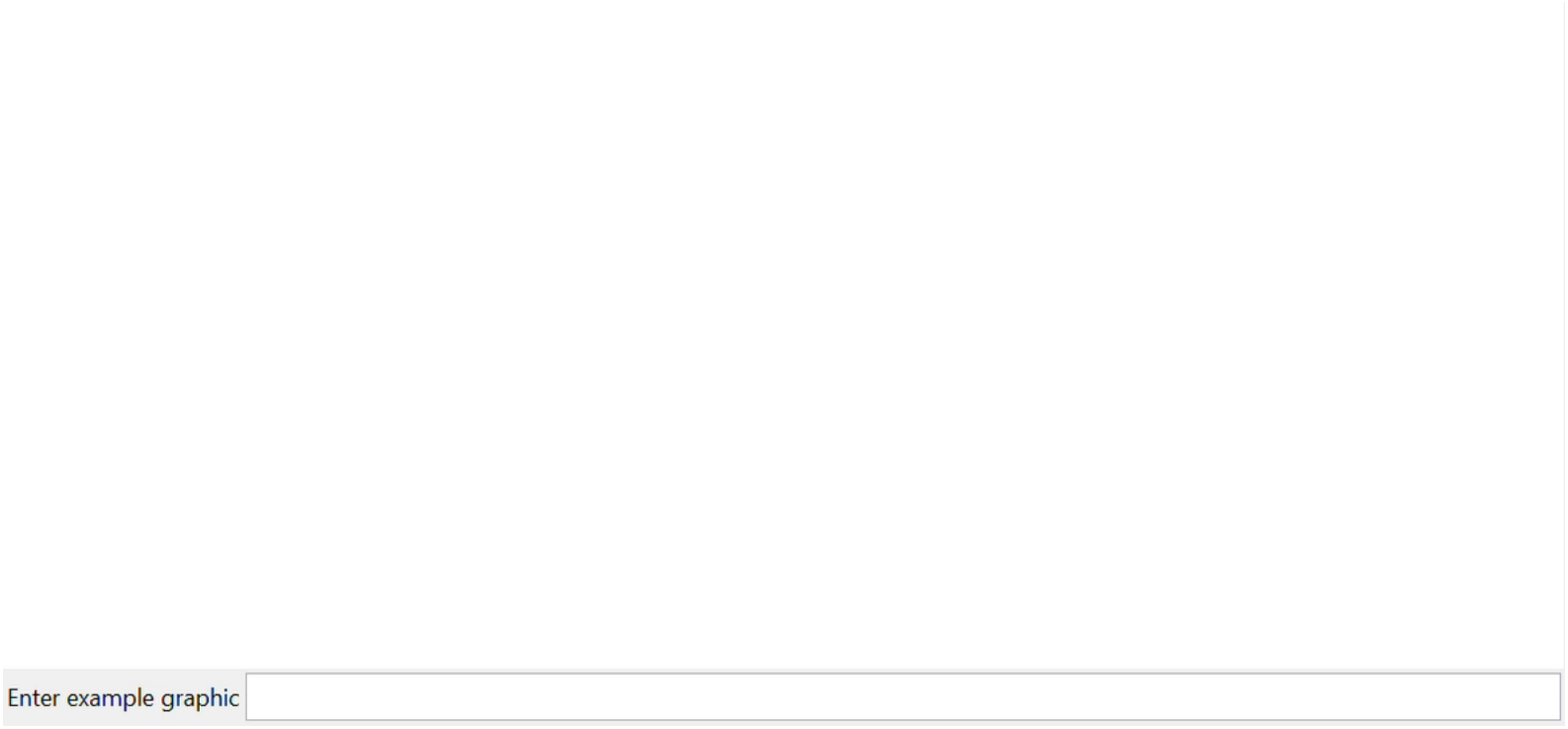
(Rosette is a framework for doing program synthesis, developed by Emina Torlak and Rastislav Bodik)

- define symbolic expression for a program
- define symbolic expression for output plant graphics from program
- define symbolic expression for # of edits between output plant graphic and plant graphic given by designer
- tell Rosette that # of edits must be less than a certain number

step 3: ask Rosette for solution program

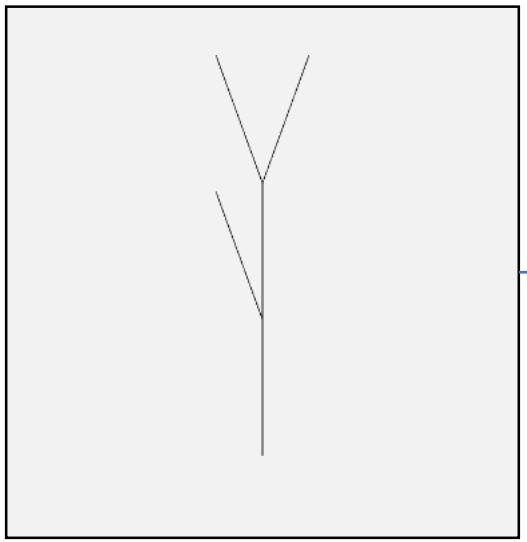
- tell Rosette to solve for program that satisfies constraint for # of edits
- Rosette compiles the constraints for a program to a Satisfiability Module Theory
- Rosette solves for a solution
- Rosette uses solution to derive expression for solution program

demo



in summary…

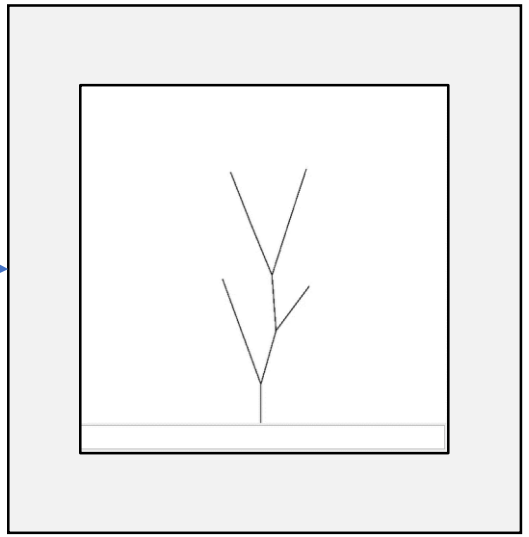
- I developed a synthesizer for programs that generate plant graphics
- given an example plant graphic, our tool will find a program that generates similar graphics



an example of a graphic that a designer wants the program to generate

```
(rule F FF)
(rule X
F[[X]+X]+F
[+FX]-X]-X)
```

a program that generates graphics (an L-system)



graphics generated by program

acknowledgements

- Professor Ras Bodik, Paul G. Allen School of Computer Science & Engineering
- Programming Languages & Software Engineering Group, Paul G. Allen School of Computer Science & Engineering