COSE 215: Theory of Computation

Undecidability (2)

Hakjoo Oh
2016 Spring

Contents

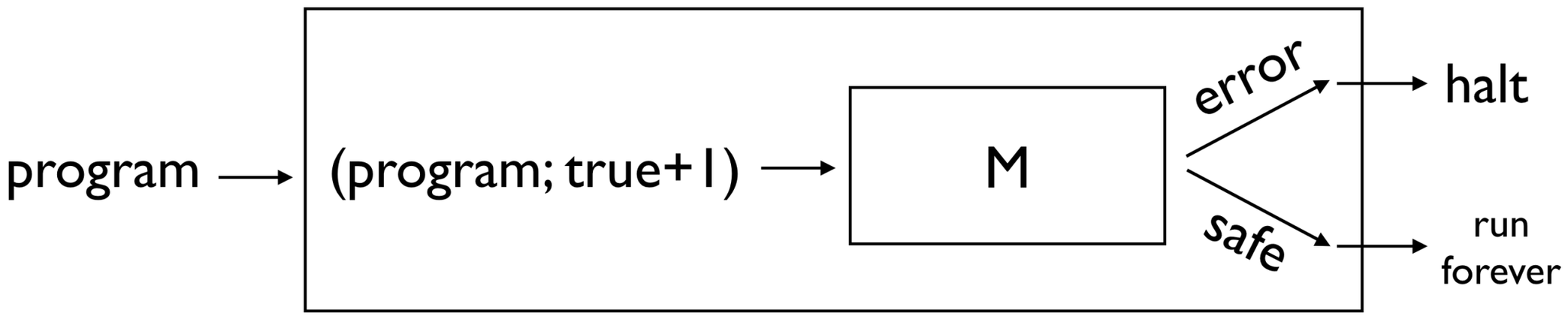
- "Real" Examples of undecidable problems
  - Halting problem
  - Program verification
  - Properties of CFGs
  - Post Correspondence Problem (PCP)
- How to deal with undecidable problems?

Halting Problem

- https://www.youtube.com/watch?v=92WHN-pAFCs

Program Verification

Program Verification



Properties of CFGs

- Is a given CFG ambiguous?
- For CFGs G_1 and G_2, is $L(G_1) \cap L(G_2) = \emptyset$?
- Is $L(G_1) = L(G_2)$?
- Is $L(G_1) = L(R)$ for some regular expression R?
- Is $L(G_1) = T^*$ for some alphabet T?

Post Correspondence Problem

- Is there a list of dominos (repetitions permitted) such that reading off top yields the string obtained by reading off bottom?

- Possible: $\{[\frac{b}{ca}], [\frac{a}{ab}], [\frac{ca}{a}], [\frac{abc}{c}]\}$

- Impossible: $\{[\frac{abc}{ab}], [\frac{ca}{a}], [\frac{acc}{ba}]\}$

Programming Technologies

- Program Analysis

- Program Synthesis

Current Technology for Safe SW

Manual, ad-hoc, postmortem:

code review, testing, simulation, debugging, etc

```
27    int i;
28    for (i = 0; i <= 12; i++) txtbuf[i] = 0;
29
30    junk = (char *)malloc(sizeof(char) * BUFLEN);
31    binbuf = (char *)malloc(sizeof(char));
32
33    if ((junk != NULL) && (binbuf != NULL)) {
34        isc_buffer_init(binbuf, junk, BUFLEN);
35        dns_name_init(dns_name, NULL);
36        dns_name_setbuffer(dns_name, binbuf);
37        result = dns_name_fromtext(dns_name,  txtbuf, NULL, 0, NULL);
38        free(junk);
39        free(binbuf);
```

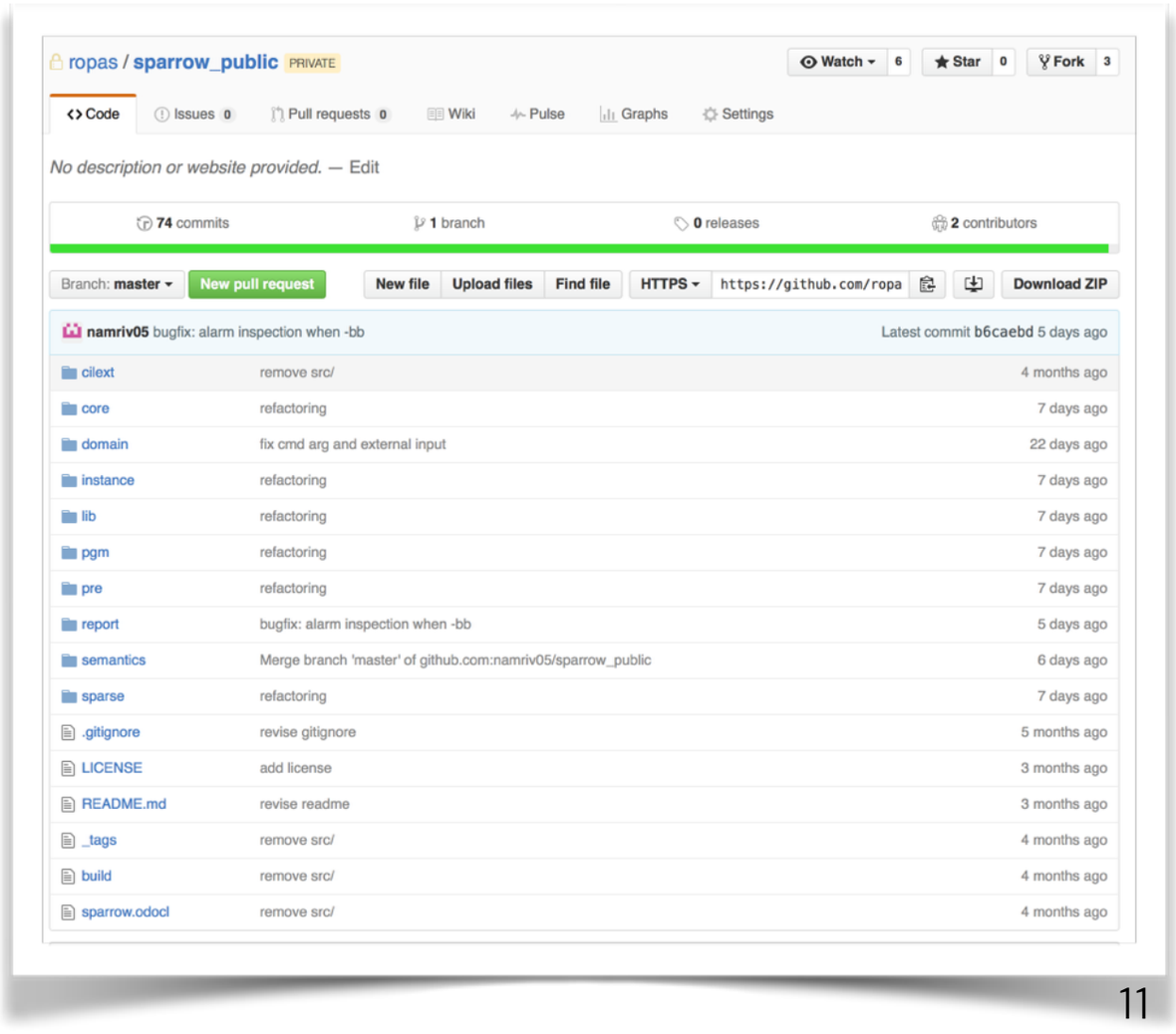
Program Analysis Technology

Technology for "Software MRI"

Sparrow

The Early Bird

- Aims to detect memory errors in C programs
  - e.g., buffer-overrun, memory leak, null-dereference, etc
- Features (vs. testing)
  - Full **automation**
  - Find bugs **early**
  - **All bugs** found



```
16      static char *curfinal = "HDACB   FE";
17
18      keysym = read_from_input ();
19
20      if ((((KeySym)(keysym) >= 0xFF9                ym) <= 0xFF94)))
21      {
22          unparseputc((char)(keysym-0xFF91 +'P'), pty);
23          key = 1;
24      }
25      else if (keysym >= 0)
26      {
27          if (keysym < 16)
28          {
29              if (read_from_input())
30              {
31                  if (keysym >= 10) return;
32                  curfinal[keysym] = 1;
33              }
34              else
35              {
36                  curfinal[keysym] = 2;
37              }
38          }
39          if (keysym < 10)
40          {
41              unparseputc(curfinal[keysym], pty);
42          }
43      }
```

curfinal: buffer of size 10

keysym: any integer

keysym: [0,15]

safe

buffer-overrun

curfinal:[10,10]
keysym: [10,15]

safe

Sparrow automatically
pinpoints the buffer-overrun bug

Static Program Analysis

- Predict SW behavior statically and automatically
  - **static**: before execution, before sell / embed
  - **automatic**: sw is analyzed by sw ("static analyzers")

- Applications
  - **bug-finding**. e.g., find runtime failures of programs
  - **security**. e.g., is this app malicious or benign?
  - **verification**. e.g., does the program meet its specification?
  - **compiler optimization**, e.g., automatic parallelization

Google

facebook

FireEye

ip Positions in Program Analysis @ Google

internship openings in 2015. Internships could

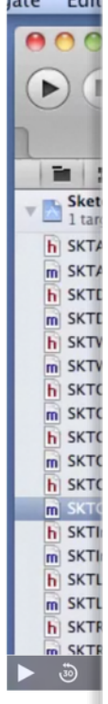
- D
- Id
- C
- R

- Ideal ca

A tool to d

2. I… ue navig
3. In the source edit
4. Use the pop-up r
5. Edit the code to f

The video shows



FireEye

Products | Solutions | Mandiant

Home › Products › Mobile Security

Mobile Security

Mobile Threat Preve

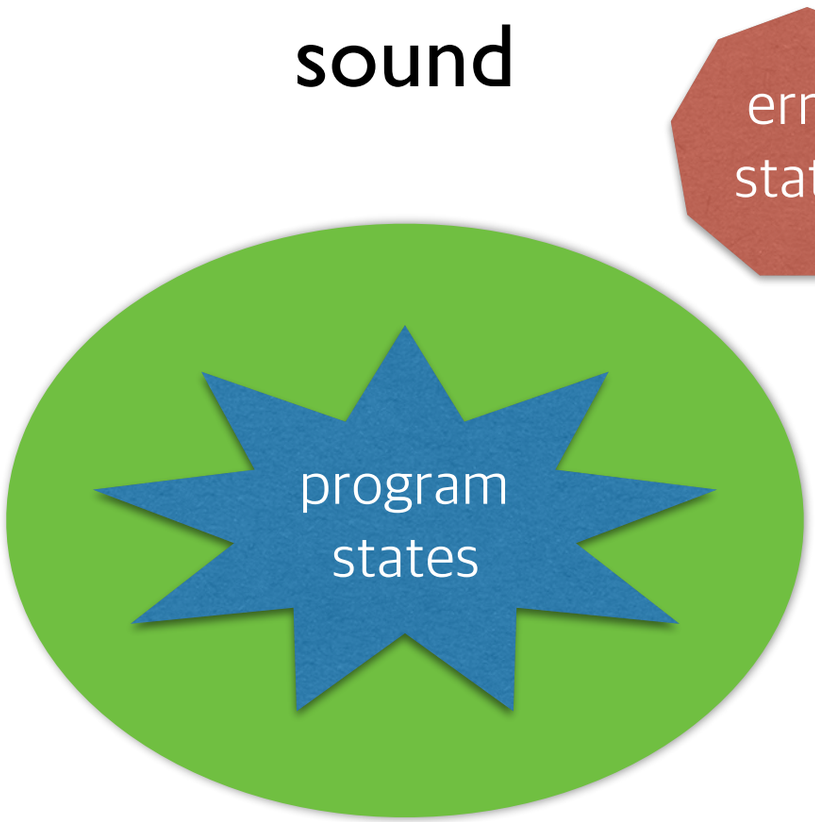
Detect and prevent cyber attacks that spy on, profile, or use mobile devices

Malicious apps compromise mobile security to access private information, such as contact lists and calendar details. They also use mobile device features, such as cameras and microphones, to spy, profile users, or conduct cyber attacks.

FireEye Mobile Security (Mobile Threat Prevention) detects and prevents these mobile threats and provides visibility into mobile security trends across the enterprise. FireEye Mobile Threat Prevention also integrates with industry leading mobile device management (MDM) providers.

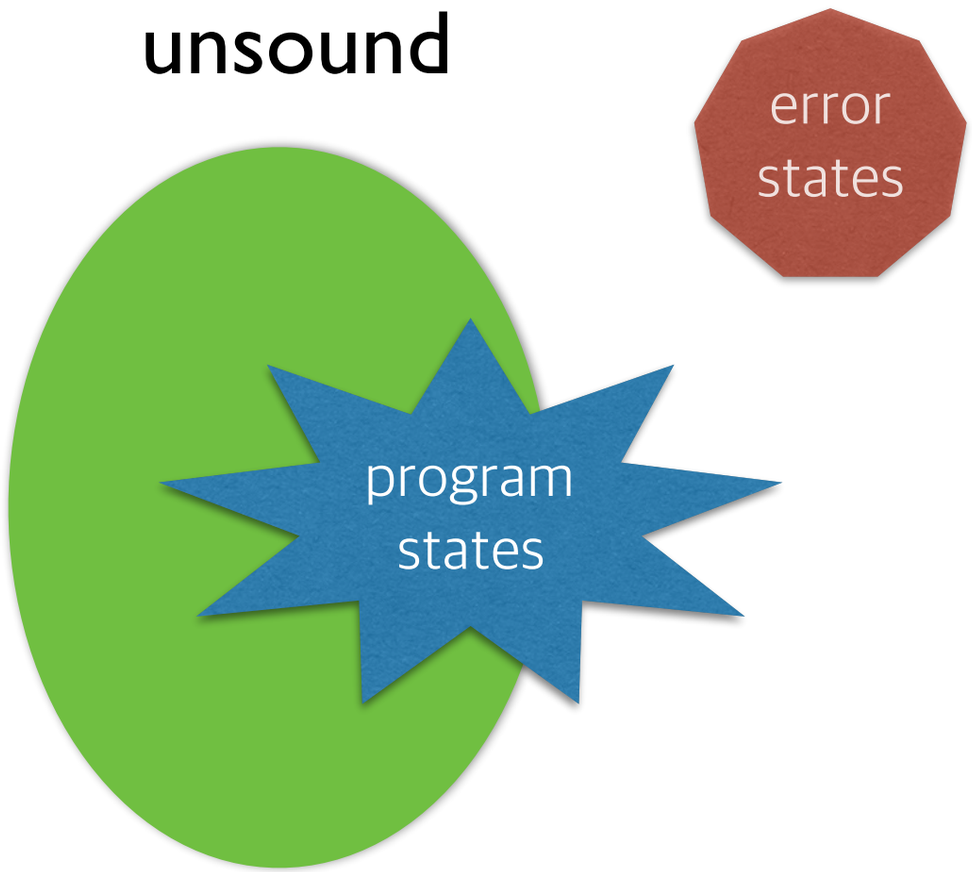
How Program Analysis Works

sound

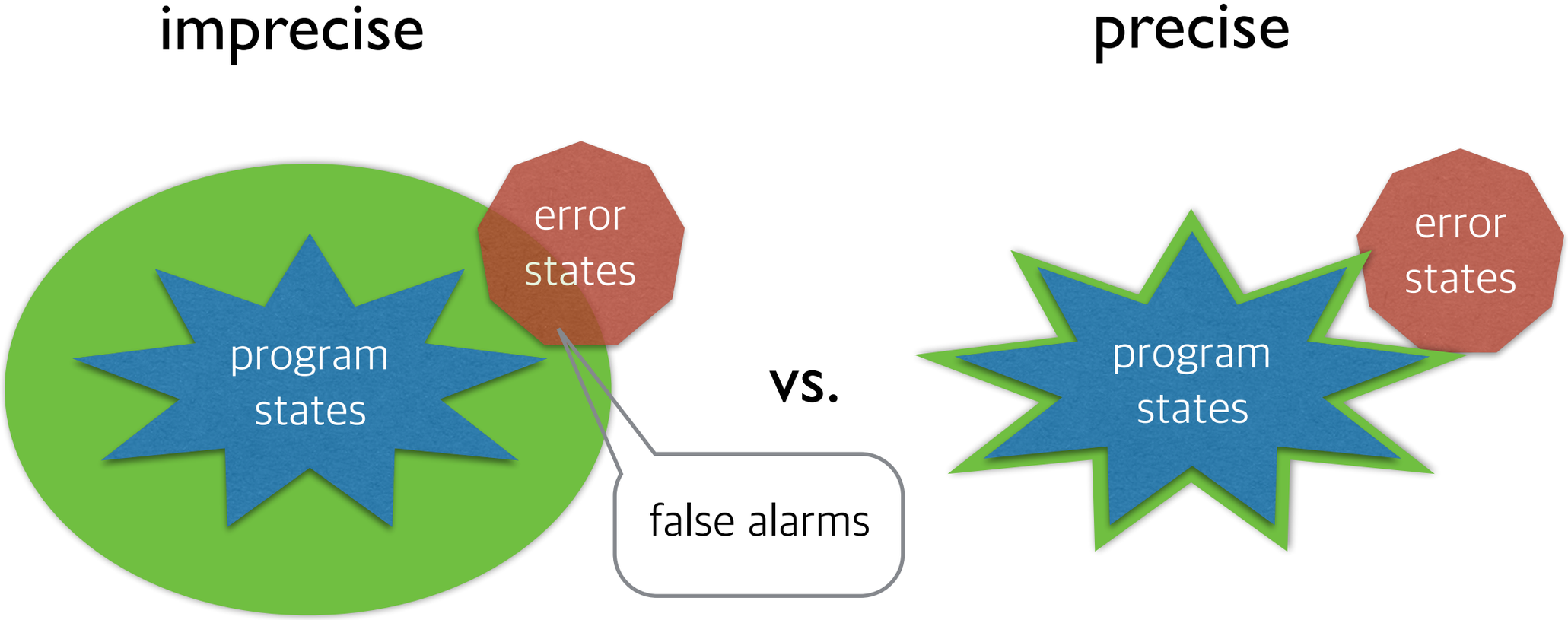


vs.

unsound



How Program Analysis Works



Program Synthesis Technology

- Currently, programs are written solely by programmers
  - programming is often repetitive, tedious, and error-prone
  - end-users are not capable of fully leveraging computational devices
- In the future, most programs will be written by programs

Program Synthesis Technology

ex) Regular Expressions

I need some way to find words that contain any combination of characters and digits but **exactly 4 digits only**, and at least one character.

EXAMPLE:

```
a1a1a1a1          // Match
1234              // NO match (no characters)
a1a1a1a1a1        // NO match
ab2b2             // NO match
cd12              // NO match
z9989             // Match
1ab26a9           // Match
1ab1c1            // NO match
12345             // NO match
24                // NO match
a2b2c2            // NO match
ab11cd            // NO match
```

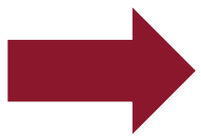
I would like to get the phone numbers from a file. I know the numbers have different forms, I can handle for a single one, but don't know how to get a uniform regex. For example

1. `xxx-xxx-xxxx`
2. `(xxx)xxx-xxxx`
3. `xxx xxx xxxx`
4. `xxxxxxxxxx`

(from stackoverflow.com)

Synthesizing Regular Expressions

Positive Examples	Negative Examples
000	0
111	00
000000	1111
010101	0101
…	…



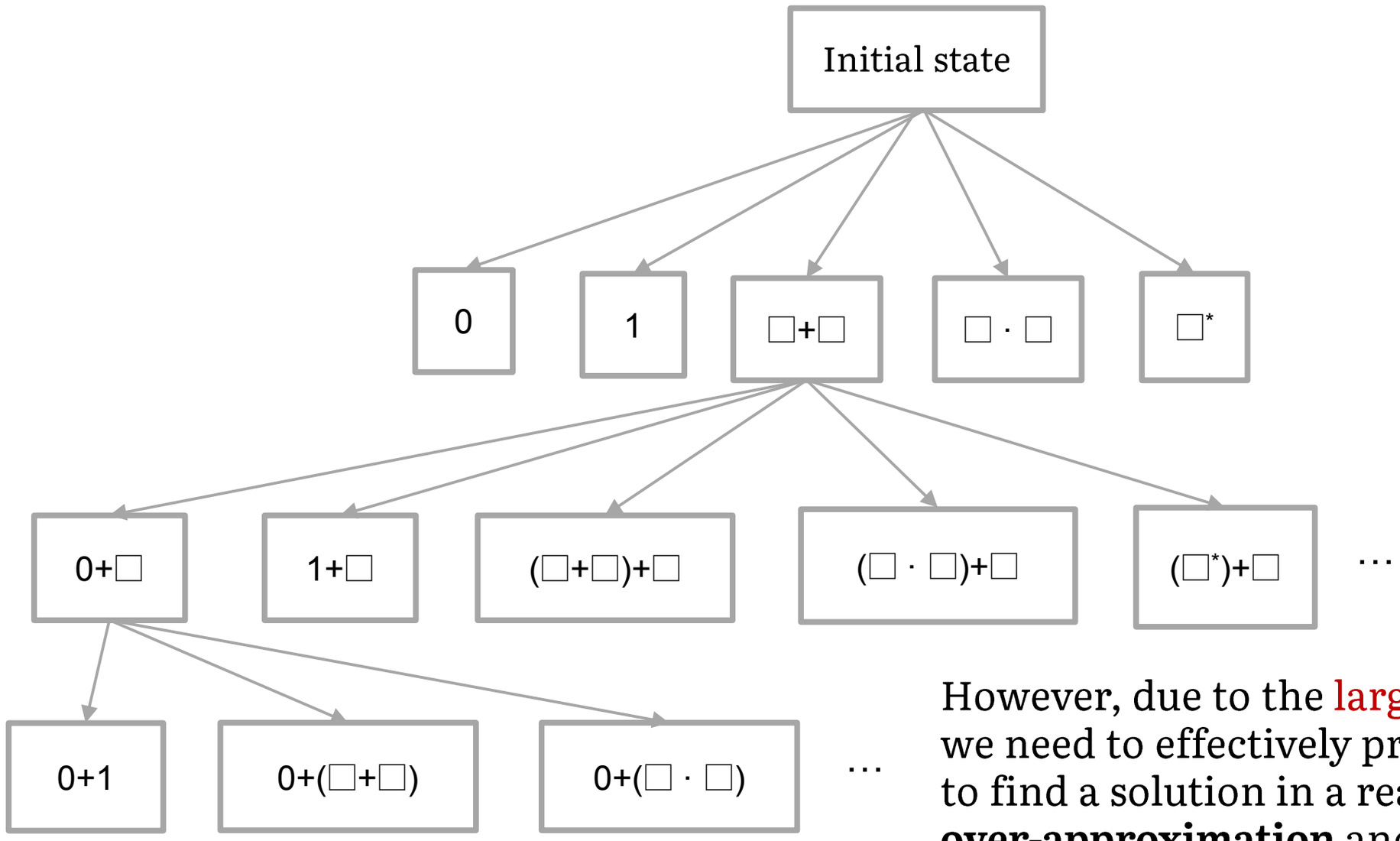
Synthesized Regular Expression

$((0 + 1)(0 + 1)(0 + 1))^*$

$L = \{w \in \{0, 1\}^* \mid \text{The length of } w \text{ is a multiple of 3}\}$

Synthesizing Regular Expressions

$$e \to a \in \Sigma \mid \varepsilon \mid \emptyset \mid e_1 + e_2 \mid e_1 \cdot e_2 \mid e^*$$



However, due to the large search space,
we need to effectively prune out the search tree
to find a solution in a reasonable time:
over-approximation and **under-approximation**

Performance

Problem	Output	Time (sec)
The length of w is a multiple of 3	((0 + 1)(0 + 1)(0 + 1))*	0.007
w begins with 1 and ends with 0	(1 (0 + 1)* 0)*	0.012
w does not end with 01	(0 + 1(0 + 1))*	0.028
w ends with 01	((0 + 1)* 01)*	0.048
w contains even number of 0s	(1 + 01*0)*	0.075
The length of w is at least 3 and its third symbol is 0	(0 + 1)(0 + 1)0(0 + 1)*	0.125
Every odd position of w is 1	1((0 + 1) + (0 + 1)1)*	0.410
n ≥ 3 and m is even	00*00(11)*	0.545
w contains the substring 0101	(0 + 1)* 0101 (0 + 1)*	1.273

FlashFill in Microsoft Excel

	A	B
1	Email	Column 2
2	Nancy.FreeHafer@fourthcoffee.com	nancy freehafer
3	Andrew.Cencici@northwindtraders.com	andrew cencici
4	Jan.Kotas@litwareinc.com	jan kotas
5	Mariya.Sergienko@gradicdesigninstitute.com	mariya sergienko
6	Steven.Thorpe@northwindtraders.com	steven thorpe
7	Michael.Neipper@northwindtraders.com	michael neipper
8	Robert.Zare@northwindtraders.com	robert zare
9	Laura.Giussani@adventure-works.com	laura giussani
10	Anne.HL@northwindtraders.com	anne hl
11	Alexander.David@contoso.com	alexander david
12	Kim.Shane@northwindtraders.com	kim shane
13	Manish.Chopra@northwindtraders.com	manish chopra
14	Gerwald.Oberleitner@northwindtraders.com	gerwald oberleitner
15	Amr.Zaki@northwindtraders.com	amr zaki
16	Yvonne.McKay@northwindtraders.com	yvonne mckay
17	Amanda.Pinto@northwindtraders.com	amanda pinto

Synthesizing Programs

```
[] ↦ []
[[1]] ↦ [[]]
[[1, 3, 5], [5, 3, 2]] ↦ [[3, 5], [5, 3]]
[[8, 4, 7, 2], [4, 6, 2, 9], [3, 4, 1, 0]] ↦
        [[8, 4, 7] [4, 6, 9], [3, 4, 1]]
```

```
dropmins x = map f x
     where f y = filter g y
           where g z = foldl h False y
                 where h t w = t || (w < z)
```

Synthesizing Data Structure Transformations from Input-Output Examples , PLDI 2016

Summary

- Many real problems are unsolvable by algorithms
- They can be approximately but usefully solved

Announcement

- Last class: 6/7 (Tuesday)

- Final exam: 6/9 (Thursday) in class