Comments on Douglas Saddy's "Perceiving and Processing Recursion in Formal Grammars"

Lee Spector
Cognitive Science
Hampshire College

Questions

- Saddy asks, "How do infants, adults and other species extract structure from a signal?"
- I ask, "What can we infer about **how** a system extracts structure from the **mere fact** that it does so?"

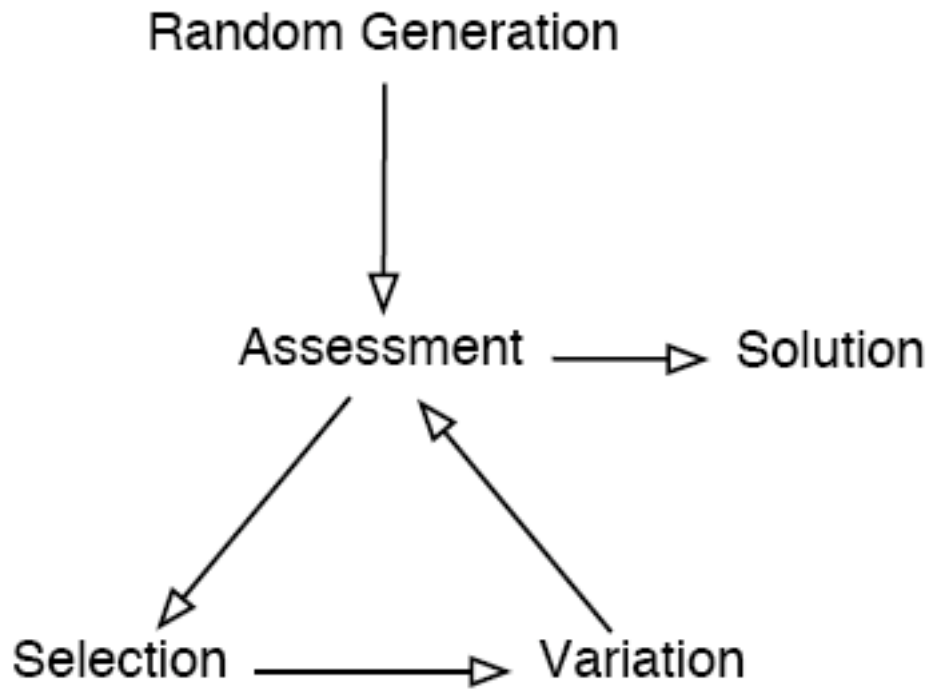
In some cases we can determine position in the Chomsky hierarchy; e.g. if the structure is $a^n b^n$ then the mechanism can't be a finite automaton.

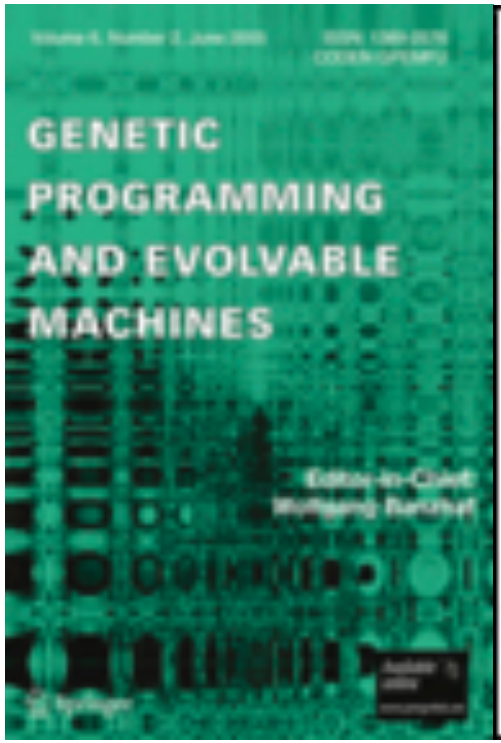
What can we determine, or even have reason to suspect, at a finer level of detail or for finite cases?

For specific cases it may be instructive to examine mechanisms that can extract the structure of interest.

How can we find such mechanisms, automatically and in general?

Genetic Programming




GENETIC
PROGRAMMING
AND EVOLVABLE
MACHINES


IEEE TRANSACTIONS ON
EVOLUTIONARY
COMPUTATION
IEEE


Evolutionary Computation


Volume 22 Number 3
Summer 2008
AIEDAM
Artificial Intelligence for Engineering Design, Analysis and Manufacturing
David C. Brown, Editor
CAMBRIDGE
UNIVERSITY PRESS

Push

- A programming language designed for programs that evolve.
- Trivial syntax:
  program → instruction | literal | (program*)
- Simplifies the evolution of programs that may use:
  - multiple data types
  - subroutines and evolved control structures
  - recursion
  - evolved evolutionary mechanisms
- Similarities to embedded pushdown automata.

Fibonacci

$$0 \rightarrow 1$$
$$1 \rightarrow 1\,0$$

(1101011011010110101101101)
(0101101011011010110110101)
(0110101101101011010110110)
(1010110110101101011011010)
(1011010110110101101011011)
(1101011011010110101101101)
(0101101011011010110110101)
(0110101101101011010110110)
(1010110101101101011011010)
(1011010110110101101011011)

Foil

$$0 \rightarrow 1$$

$$1 \rightarrow 1\,0\,1$$

(1110110110111011011101101)
(0111011011011101101110110)
(1011101101101110110111011)
(1101110110110111011011101)
(0110111011011011101101110)
(1011011011101101110110111)
(1101101101110110111011011)
(1110110110111011011101101)
(0111011011011101101110110)
(1011101101101110110111011)

Evolved Discriminator

```
((INTEGER.MIN CODE.FROMNAME)
 ((CODE.IF
   (EXEC.DO*RANGE
    (CODE.IF) EXEC.IF) INTEGER.%
   (EXEC.=  (INTEGER.ROT)
            (INTEGER.SHOVE INTEGER.MAX FLOAT.SHOVE EXEC.YANK EXEC.K))
   CODE.NTH)
  (CODE.SWAP
   CODE.SUBST
   (INTEGER.ROT CODE.YANKDUP INTEGER.ROT) CODE.DO*)
  ((CODE.NULL
    (BOOLEAN.= (FLOAT.FROMBOOLEAN CODE.CAR)))
   INTEGER.DEFINE)
  CODE.SIZE)
 (NAME.ROT (CODE.DEFINE
            CODE.DO*COUNT BOOLEAN.NOT
            ((INTEGER./ (EXEC.= INTEGER.YANKDUP BOOLEAN.SHOVE))
             (EXEC.ROT CODE.YANK) CODE.DO*TIMES)
            (CODE.DO*TIMES))
           FLOAT.=)
 FLOAT.FROMBOOLEAN
 (CODE.MEMBER ((EXEC.SHOVE) -3)
              (FLOAT.FLUSH) BOOLEAN.=) CODE.MEMBER)
```

Simplified

```
(EXEC.IF x FALSE CODE.DO* BOOLEAN.POP
 BOOLEAN.POP BOOLEAN.NOT BOOLEAN.=)
```

CODE.DO* is a recursive operation.

Repeat Lengths

Fibonacci	Foil
(21112121112111121211)	(312121312131211)
(1112111212111212111)	(131212131213121)
(12111212111211112121)	(113121213121312)
(11112121112111212111)	(213121213121311)
(1121112121112111212)	(121312121312131)
(21112121112111121211)	(112121312131213)
(1112111212111212111)	(212121312131212)
(12111212111211112121)	(312121312131211)
(11112111212111212111)	(131212131213121)
(1121112121112111212)	(113121213121312)

Repeat Length Repeat Lengths

Fibonacci

(131113131112)
(31311131114)
(1131113131111)
(41113131113)
(213111313111)
(131113131112)
(31311131114)
(1131113131111)
(41311131113)
(213111313111)

Foil

(1111111111112)
(11111111111111)
(2111111111111)
(1111111111112)
(11111111111111)
(2111111111111)
(11111111111111)
(1111111111112)
(11111111111111)
(2111111111111)

The discrimination can be made on the basis of relatively simple features; full understanding of the structure is not required.

There are probably many such features, and many ways to compute each of them.

Are there discriminators that don't employ recursion?

Discriminator evolved with restricted instruction set (simplified)

```
(EXEC.IF (BOOLEAN.POP BOOLEAN.NOT)
 BOOLEAN.NOT EXEC.IF X BOOLEAN.=
 BOOLEAN.AND EXEC.IF X BOOLEAN.POP
 BOOLEAN.OR BOOLEAN.ROT BOOLEAN.ROT
 BOOLEAN.SWAP BOOLEAN.NOT BOOLEAN.OR
 BOOLEAN.NOT)
```

- It's bigger.
- Does it generalize?

Implications

- Inferring mechanism from behavior is a tricky business.
- Recursion is negotiable (Stabler).
- We might gain new insights by exploring the space of mechanisms that can extract particular kinds of structure.
- Genetic programming allows us to do this.

Question for Saddy

What can we really infer about **how** your subjects extracted the structure that they extracted from the mere fact that they did so?