

# Flight of the FINCH through the Java Wilderness

**Michael Orlov and Moshe Sipper**

[orlov](mailto:orlov@cs.bgu.ac.il), [sipper@cs.bgu.ac.il](mailto:sipper@cs.bgu.ac.il)

Department of Computer Science  
Ben-Gurion University, Israel



2010 HUMIES  
July 9, GECCO, Portland

















Evolved program computing  $x^4 + x^3 + x^2 + x$ :

```
class SimpleSymbolicRegression_0_7199 {
    Number simpleRegression(Number num) {
        double d = num.doubleValue();
        d = num.doubleValue();
        double d1 = d; d = Double.valueOf(d + d * d *
            num.doubleValue()).doubleValue();
        return Double.valueOf(d +
            (d = num.doubleValue()) * num.doubleValue());
    }

    /* Rest of class unchanged */
}
```



Evolved program computing  $x^9 + x^8 + \dots + x^2 + x$ :

```
Number simpleRegression(Number num) {  
    double d = num.doubleValue();  
    return Double.valueOf(d + (d * (d * (d +  
        ((d = num.doubleValue()) +  
            ((num.doubleValue() * (d = d) + d) *  
                d + d) * d + d) * d)  
        * d) + d) + d) * d);  
}
```



## Interlude: Important Note



Flight of the  
Finch through  
the Java  
Wilderness

Michael Orlov  
Moshe Sipper

We're not simply solving Ants or Spirals again (yawn).

We're taking a **bad program** and **improving** it,  
outputting in the end a **perfect** program.

Thus, we're solving one of the most complex problems known  
to man. Indeed, we're solving a problem that currently falls  
exclusively within the domain of humans—  
**machines don't do this.**



## Evolved program solving Intertwined Spirals problem:

uses sign of  $\sin\left(\frac{9}{4}\pi^2\sqrt{x^2+y^2} - \tan^{-1}\frac{y}{x}\right)$

```
boolean isFirst(double x, double y) {
    double a, b, c, e;
    a = Math.hypot(x, y); e = y;
    c = Math.atan2(y, b = x) +
        -(b = Math.atan2(a, -a))
        * (c = a + a) * (b + (c = b));
    e = -b * Math.sin(c);
    if (e < -0.0056126487018762772) {
        b = Math.atan2(a, -a);
        b = Math.atan2(a * c + b, x); b = x;
        return false;
    }
    else
        return true;
}
```



## Compare with Koza's best-of-run:

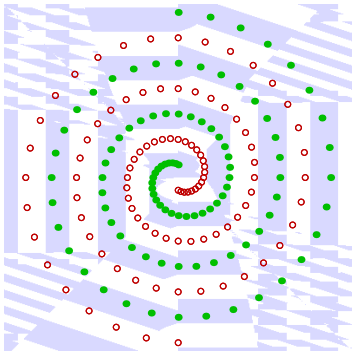
```
(sin (iflte (iflte (+ Y Y) (+ X Y) (- X Y) (+ Y Y)) (* X X)
(sin (iflte (% Y Y) (% (sin (sin (% Y 0.30400002)))) X) (% Y
0.30400002) (iflte (iflte (% (sin (% (% Y (+ X Y))
0.30400002)) (+ X Y)) (% X -0.10399997) (- X Y) (* (+
-0.12499994 -0.15999997) (- X Y))) 0.30400002 (sin (sin
(iflte (% (sin (% (% Y 0.30400002) 0.30400002)) (+ X Y))
(% (sin Y) Y) (sin (sin (sin (% (sin X) (+ -0.12499994
-0.15999997)))))) (% (+ (+ X Y) (+ Y Y)) 0.30400002))))
(+ (+ X Y) (+ Y Y)))) (sin (iflte (iflte Y (+ X Y) (- X Y)
(+ Y Y)) (* X X) (sin (iflte (% Y Y) (% (sin (sin (% Y
0.30400002)))) X) (% Y 0.30400002) (sin (sin (iflte (iflte
(sin (% (sin X) (+ -0.12499994 -0.15999997)))) (% X
-0.10399997) (- X Y) (+ X Y)) (sin (% (sin X) (+
-0.12499994 -0.15999997)))) (sin (sin (% (sin X) (+
-0.12499994 -0.15999997)))) (+ (+ X Y) (+ Y Y)))))) (%
Y 0.30400002))))))
```

# Tree GP vs. FINCH

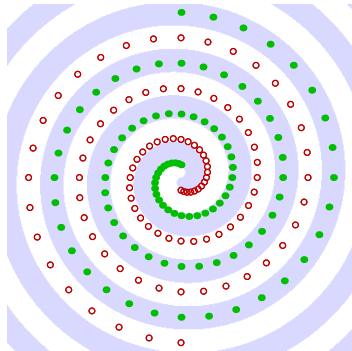


And compare the phenotypes:

Koza's:



Ours:



Flight of the  
Finch through  
the Java  
Wilderness

Michael Orlov  
Moshe Sipper



### Evolved program computing sum of values in array: (loop solution)

```
public int sumlist(int list[]) {  
    int sum = 0;  
    int size = list.length;  
    for (int tmp = 0; tmp < list.length; tmp++) {  
        size = tmp;  
        sum = sum - (0 - list[tmp]);  
    }  
    return sum;  
}
```

Yes, FINCH can handle loops...



### Evolved program computing sum of values in array: (List solution)

```
int sumlist(List list) {
    int sum = 0;
    int size = list.size();
    for (Iterator iterator = list.iterator();
         iterator.hasNext(); ) {
        int tmp = ((Integer) iterator.next())
                 .intValue();

        tmp = tmp + sum;
        if (tmp == list.size() + sum)
            sum = tmp;
        sum = tmp;
    }
    return sum;
}
```

Yes, FINCH can handle Java 5.0 constructs. . .









# The Insidious Errors



```
1 int negamaxAB(TicTacToeBoard board,
2     int alpha, int beta, boolean save) {
3     Position[] free = getFreeCells(board);
4     // utility is derived from the number of free cells left
5     if (board.getWinner() != null)
6         alpha = utility(board, free);
7     else if (free.length == 0)
8         alpha = 0; save = false;
9     else for (Position move: free) {
10        TicTacToeBoard copy = board.clone();
11        copy.play(move.row(), move.col(),
12            copy.getTurn());
13        int utility = - (removed) negamaxAB(copy,
14            -beta, -alpha, false save );
15        if (utility > alpha) {
16            alpha = utility;
17            if (save)
18                // save the move into a class instance field
19                chosenMove = move;
20            if (alpha >= -beta beta >= alpha )
21                break;
22        }
23    }
24    return alpha;
25 }
```

Flight of the  
Finch through  
the Java  
Wilderness

Michael Orlov  
Moshe Sipper



**(G) The result solves a problem of indisputable difficulty in its field.**

- There is a widely recognized need for automatic improvement of existing software.
- Improving software is indisputably difficult (G) (Did anybody say 'difficult'? Merely 'difficult'?)
- No technique previously existed that allowed the automatic improvement of **unrestricted** software written in a **widely used**, real-life programming language.
- And along came FINCH...







## Why is Result Best? (cont'd)



Flight of the  
Finch through  
the Java  
Wilderness

Michael Orlov  
Moshe Sipper

- We have taken another major step forward.
- Ours is the first approach that allows viable evolution of extant, real-world software in a mainstream programming language (Java is one of the 2 most popular programming languages).
- Moreover, FINCH is not limited to Java: Scala, Groovy, Jython, Kawa, JavaFX Script, Clojure.

## Why is Result Best? (cont'd)



Flight of the  
Finch through  
the Java  
Wilderness

Michael Orlov  
Moshe Sipper

- “Judiciously used, digital evolution can substantially augment the cognitive limits of human designers and can find novel (possibly counterintuitive) solutions to complex . . . system design problems.”

(Recent study by US DoD on futuristic systems)

- FINCH represents a significant step on the (long) path toward full-fledged **Darwinian Software Development**.





“I believe that in about fifty years’ time it will be possible, to programme computers . . . to make them play the imitation game so well that an average interrogator will not have more than 70 per cent. chance of making the right identification after five minutes of questioning.”

A. M. Turing, “Computing machinery and intelligence,”  
*Mind*, 59(236), 433-460, Oct. 1950



“... despite its current widespread use, there was, within living memory, equal skepticism about whether compiled code could be trusted. If a similar change of attitude to evolved code occurs over time ...”

*M. Harman, “Automated patching techniques: The fix is in,”  
Communications of the ACM, 53(5), 108, May 2010*



We believe that in about fifty years' time it will be possible, to program computers. . . by means of evolution.

Not merely **possible** but indeed **prevalent**.

Turing was wrong—will we be?



We believe that in about fifty years' time it will be possible, to program computers. . . by means of evolution.

Not merely **possible** but indeed **prevalent**.

Turing was wrong—will we be?



We believe that in about fifty years' time it will be possible, to program computers. . . by means of evolution.

Not merely **possible** but indeed **prevalent**.

Turing was wrong—will we be?

# 2060 (cont'd)



Flight of the  
Finch through  
the Java  
Wilderness

Michael Orlov  
Moshe Sipper

To find out, please register for GECCO 2060.



