



Recurrent Cartesian Genetic Programming Applied to Famous Mathematical Sequences

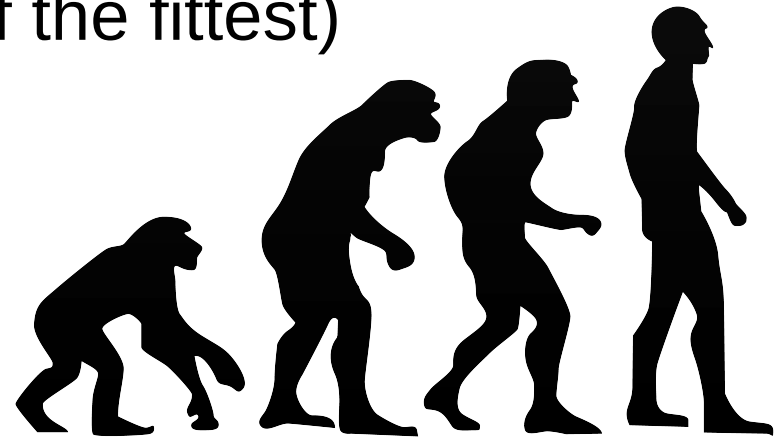
Andy Turner





Intro to Evolutionary Algorithms

- Evolutionary Algorithms are a search heuristic which take inspiration from Biological Evolution.
- It has concepts such as:
 - Parents and Children
 - Generations
 - Natural Selection (survival of the fittest)
 - Reproduction





1. Create a population of random solutions
2. **while** (good solution not found) **do**
3. determine suitability of each solution
4. select best solutions as parents
5. create new solutions based on parents
6. **end while**



Intro to Evolutionary Algorithms

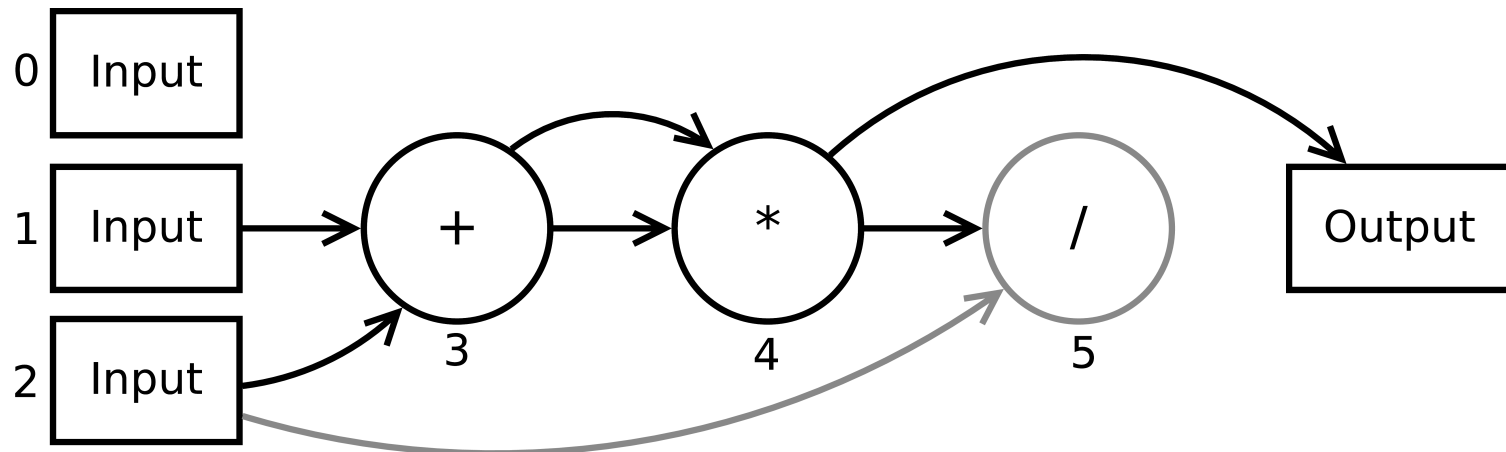
- Imagine you were playing **mastermind**
- Solution => a set of four pins
- Random population => choosing random pins
- Solution suitability => score of chosen pins
- Selection => choose highest scoring pins
- Reproduction => copy pins and change random pin





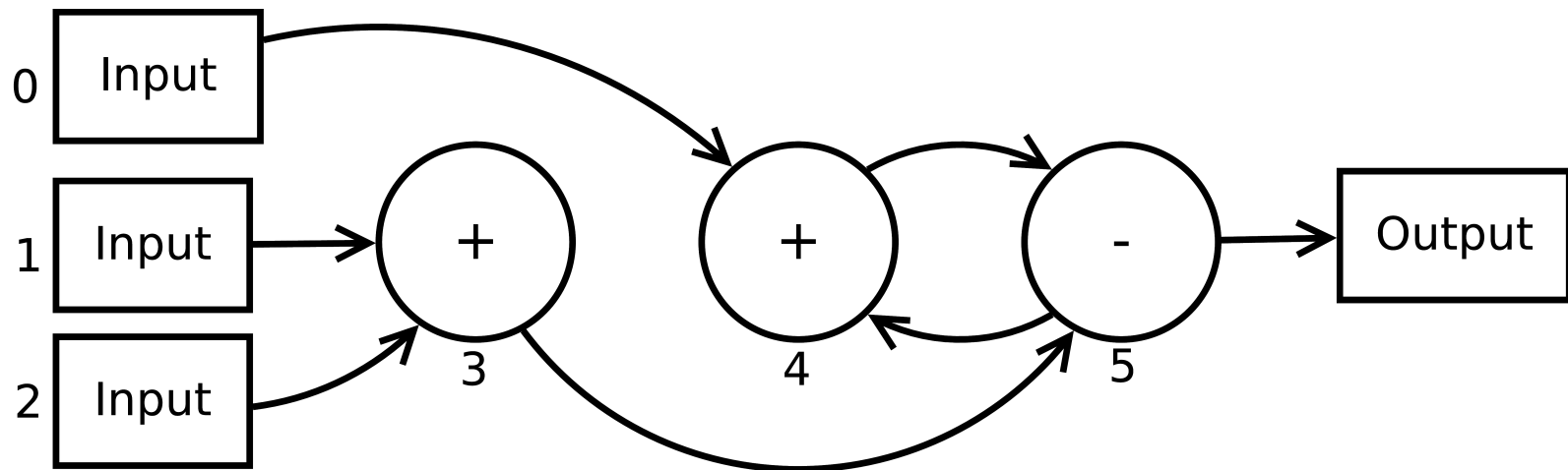
Cartesian Genetic Programming

- The solutions describe graph based programs
- The programs are arranged a graphs
 - Inputs
 - Outputs
 - Computational nodes





- Enables programs with recurrent connections
- Solutions can contain memory / feedback
- Solutions can be recursive
- Wider range of applications now possible





- Explicit Equations are of the form:

$$y(x) = x^2$$

- Recurrent Equations are of the form:

$$y(x) = x + y(x-1)$$

- All explicit equations have a recurrent form
- Not all recurrent equations have an explicit form

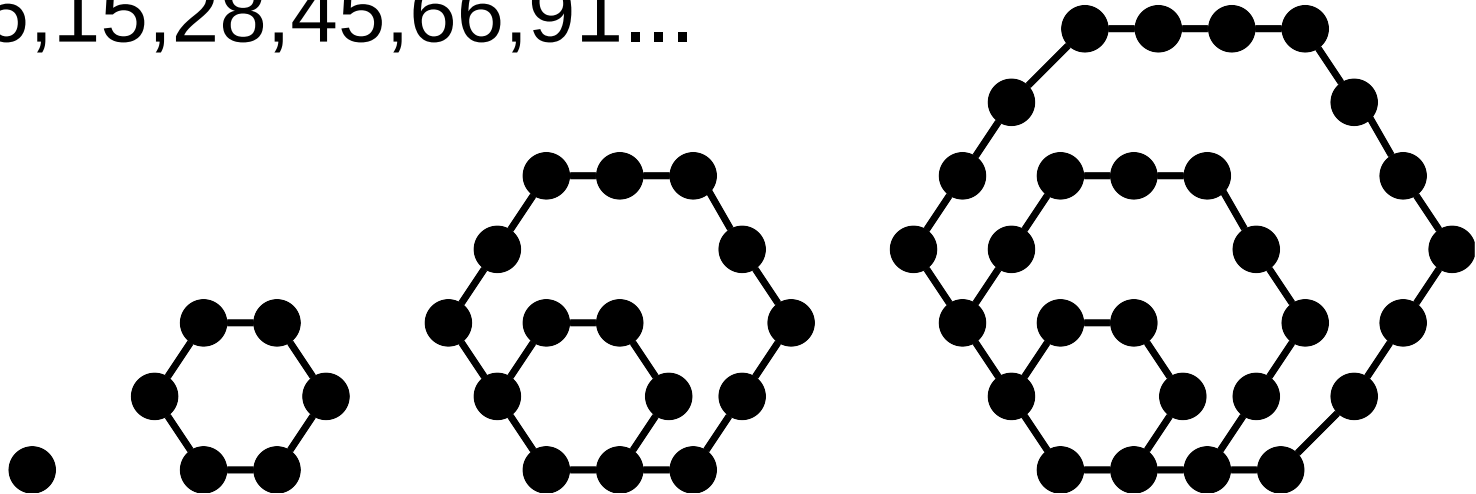


- It is known that CGP can be used to create explicit equations.
- But can RCGP be used to create recurrent equations?
- And is this worth doing?



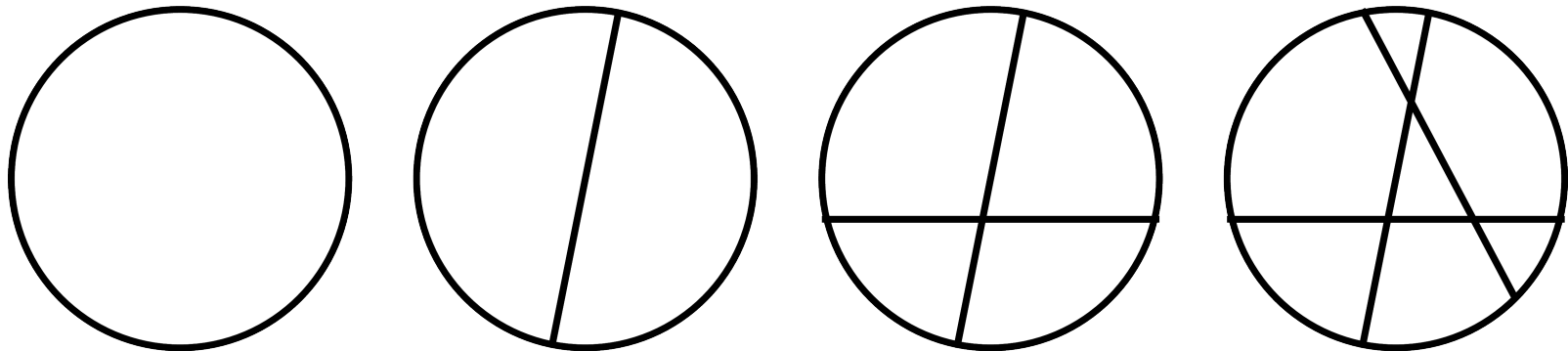
- Producing Number Sequences
 - “A sequence, in mathematics, is a string of objects, like numbers, that follow a particular pattern.” - *internet*
- CGP explicit solutions:
 - Should output the n^{th} value in the sequence when the input n is applied.
- RCGP recurrent solutions:
 - Input fixed at 1
 - Should output the n^{th} value in the sequence on the n^{th} iteration of the equation.

- Number of dots which make up a sequence of hexagons.
- 1, 6, 15, 28, 45, 66, 91...



$$y(n) = \frac{2n(2n - 1)}{2}$$

- The Lazy Caterers Sequence is the number of slices a cake can be divided into with n cuts.
- 1, 2, 4, 7, 11, 16, 22, 29, ...



$$y(n) = \frac{n^2 + n + 2}{2}$$

- The sequence of Magic Constants are the values each row, column and diagonal of a $n \times n$ magic square can sum to.
- 1, 5, 15, 34, 65, 111, ...

2	7	6
9	5	1
4	3	8

7	12	1	14
2	13	8	11
16	3	10	5
9	6	15	4

23	6	19	2	15
4	12	25	8	16
10	18	1	14	22
11	24	7	20	3
17	5	13	21	9

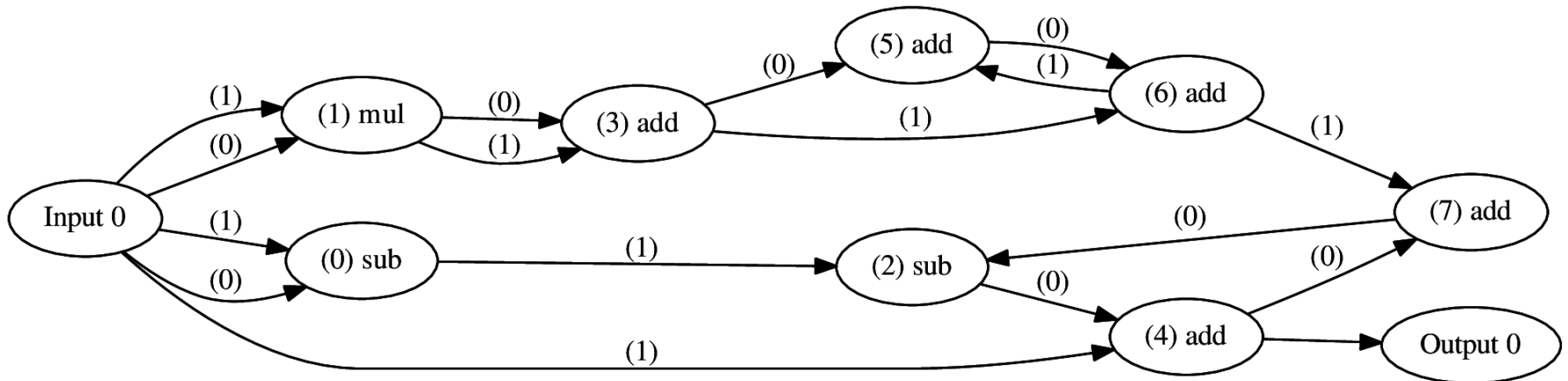
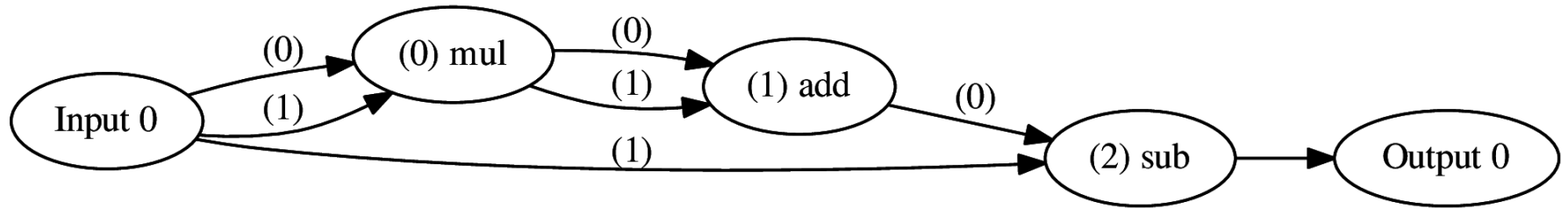
$$y(n) = \frac{n(n^2 + 1)}{2}$$

- The Fibonacci sequence is such that each value is the sum of the previous two values; where the first two values are set as one.
- 1,1,2,3,5,8,13,21,...

$$y(n) = \begin{cases} 1, & \text{if } n \leq 1 \\ y(n-1) + y(n-2), & \text{otherwise} \end{cases}$$

$$y(n) = \frac{(1 + \sqrt{5})^n - (1 - \sqrt{5})^n}{2^n \sqrt{5}}$$

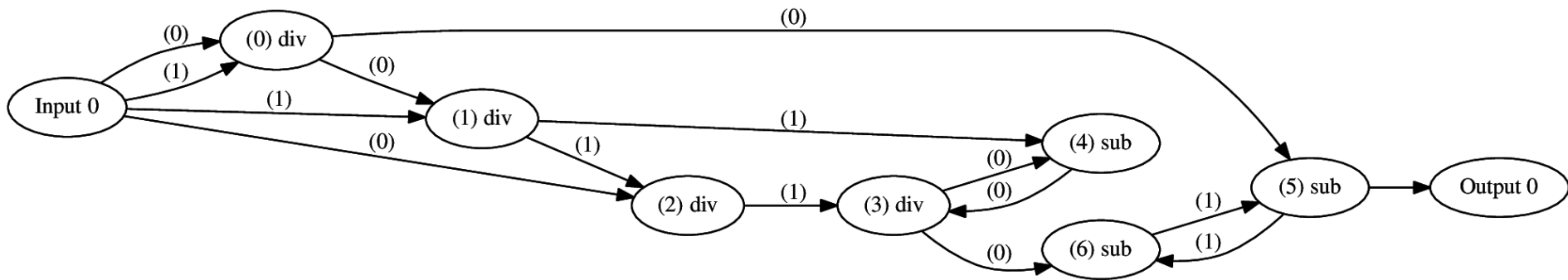
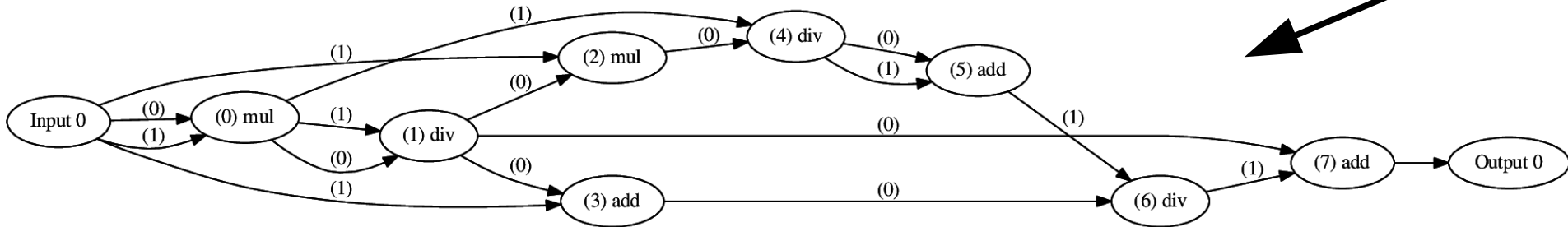
Hexagonal Numbers





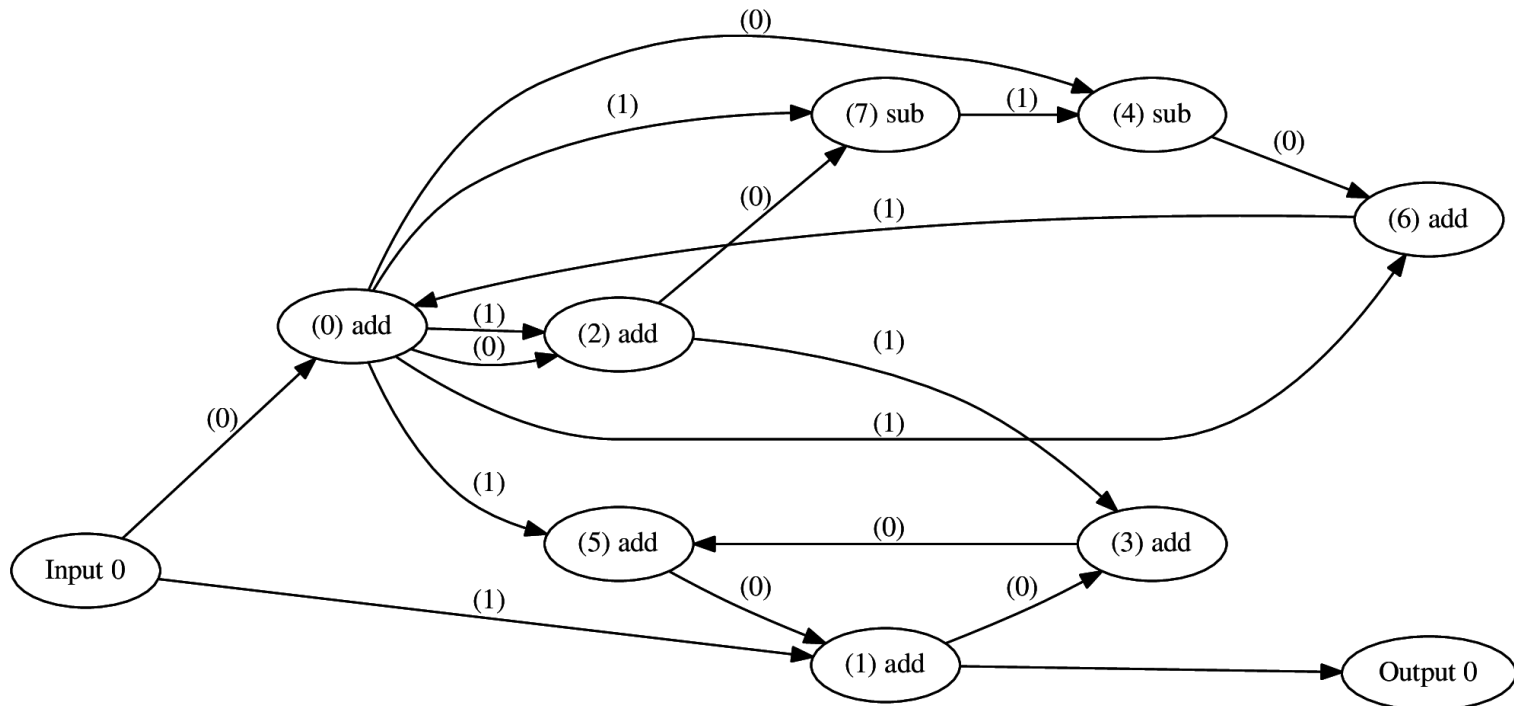
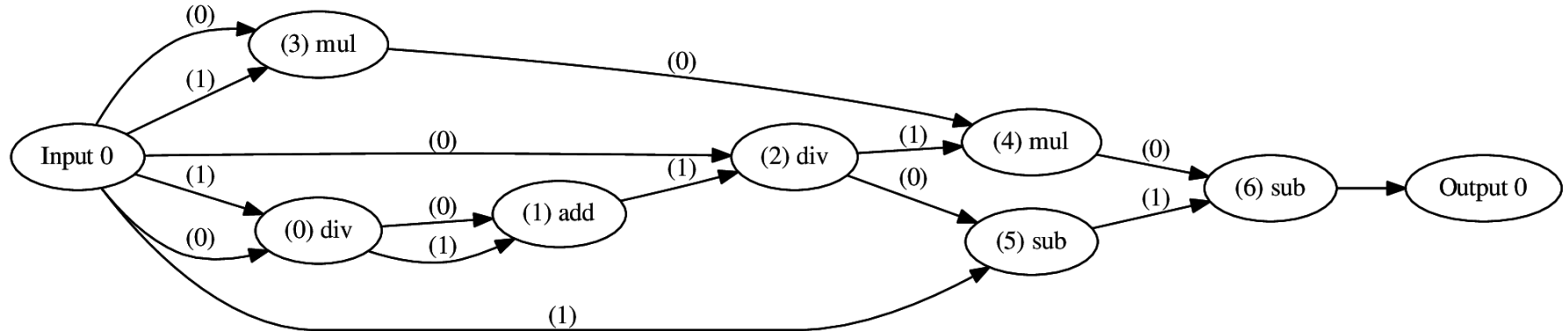
Lazy Caterers

Struggled

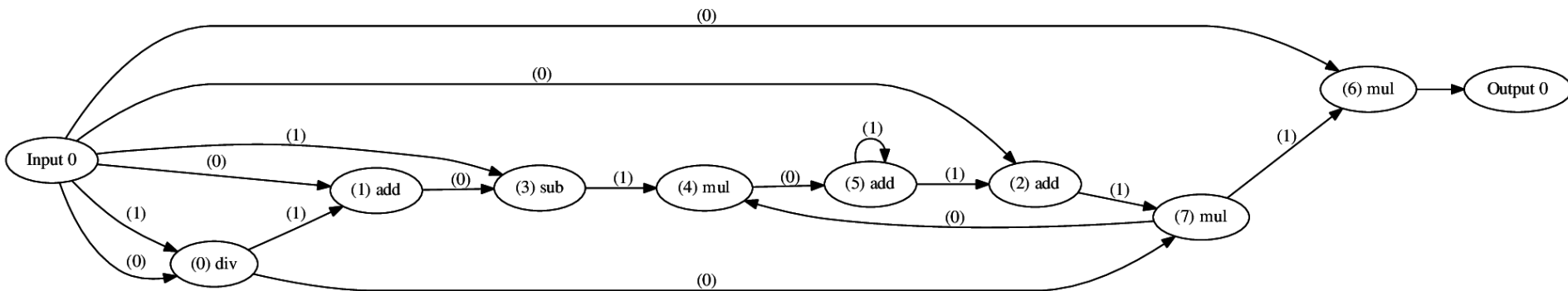




Magic Constants



- CGP could not find an explicit solution...



■ Other methods on the Fibonacci Sequence*

Method	Evaluations
RCGP	27,075
Multi-niche Genetic Programming	200,000
Probabilistic Adaptive Mapping Developmental GP	212,000
SMCGP	1,000,000
Machine Language Programs	1,000,000
Object Oriented GP	20,000,000

*The experimental set-up was slightly different between each of these results and so can only be used a rough comparison.

- RCGP found solutions to **all** given sequences.
- CGP **cannot** find solutions to tasks which require recurrent solutions.
- RCGP appears to be a **competitive** method for creating recurrent solutions.
- Therefore RCGP is an important extension to CGP.
- (RCGP is also capable of creating explicit solutions.)



- **CGP Library**
- <http://cgplibrary.co.uk/>
- Cross Platform (Linux, Windows, Mac)
- Open Source (GNU LGPL)
- Full Documentation with Tutorials
- Very Simple
- Written in C (fast)
- Language Bindings (soon)



github
SOCIAL CODING





Benchmark	CGP		RCGP	
	Successful	Avg Evaluations	Successful	Avg Evaluations
Hexagonal	50/50	2,481	50/50	39,279
Lazy Caterers	0/50	-	48/50	7,626
Magic Constants	50/50	557,592	43/50	686,929
Fibonacci	0/50	-	50/50	27,075

- Recurrent CGP always found a solution (but not always the quickest)
- CGP failed to find a solution to the Fibonacci task (unsurprising)
- CGP failed to find a solution to the Lazy Caterers task (surprising)

Parameter	Value
Max Evaluations	4,000,000
Max Nodes	20
Node Arity	2
Mutation Type	Probabilistic
Mutation Rate	5%
Recurrent Connection Probability	10%
Function Set	add, sub, mul, div