Evolving Custom Communication Protocols

by Wes Faler of Part-Time Scientists
for 28C3, Berlin Germany, 2011
wf@ptscientists.com

- Why
- What
- How
- Code and Details
- Results
- Your turn!

<presentation>

- Part-Time Scientists needed a communication protocol between our Earth stations and our rover on the moon.

Why

What

- Given:
  - A fixed set of hardware components for a packet transmitter and receiver.
  - Video and telemetry streams that must go from the moon to Earth.
  - A stream of commands that must go from Earth to the rover on the moon.
- Create:
  - A finite state machine that provides the network and application layer control of a transceiver used on both earth and the moon.

- Must be a published open protocol.
- Cannot use encryption.
- Long latency.
- High packet cost and value.
- Bandwidth is limited and saturated.
- May tunnel through another protocol.
- TCP/IP is not an option.

What – Complications

- $Design_N = Debate_N\,[Design_{N-1}] + \alpha\ Debate_{N-1}[Design_{N-1}]$

Easy as 1…

Easy as 1…2…

- Create parameterized algorithm
- One-at-time simulation

- Parameter sweep

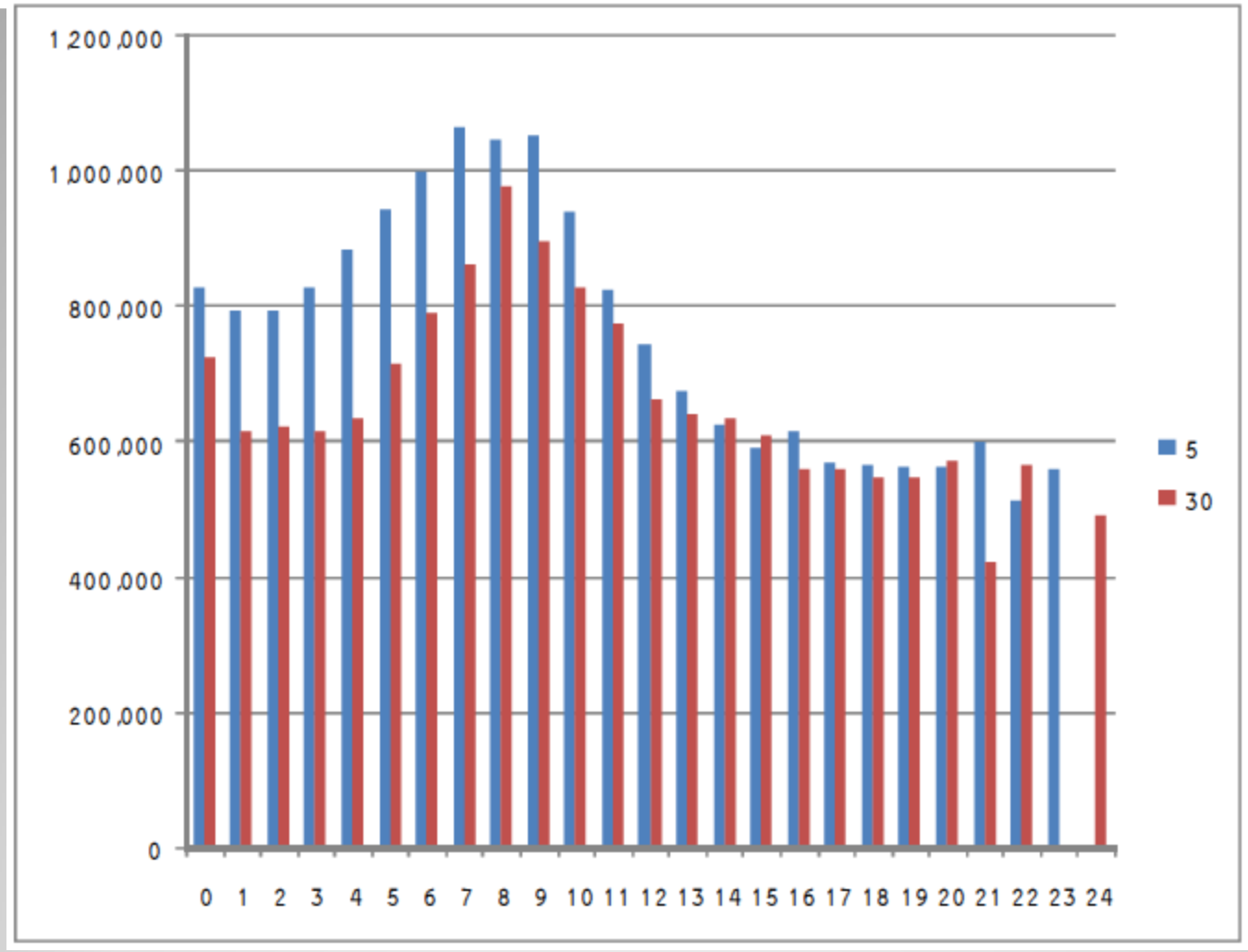
```
int driver_movesPerTransaction[] = { 1, 20 };
int driver_wheelTicksPerCommand[] = { 512, 4096 };
int driver_commandsPerPacket[] = { 1, 25 };
int driver_moveCommandsPerCameraCommand[] = { 0, 10 };
float earth_packetTimeout[] = { 5, 30 };
int earth_maxAllowedBacklogCount[] = { 10 };
float burst_upstreamBurstErrorChance[] = { 0, 0.05 };
float burst_downstreamBurstErrorChance[] = { 0, 0.05 };
float rover_naggleDuration[] = { 0.5 };
int rover_packetQueueSize[] = { 10, 0 };
int rover_shouldFinishMoveQueueBeforeCommRecovery[] = { 1, 0 };
float rover_commRecoveryDuration[] = { 5, 30 };
float rover_moveCommandCommOutageChance[] = { 0.01, 0 };
float rover_steerCommandCommOutageChance[] = { 0.01, 0 };
```

Easy as 1…2…3…

- GPU

- Before: 1 simulation/second
- Port unoptimized C++ code to GPU
  - Least programming effort possible
  - Just for parameter sweeps
- After: 700 seconds for 5200 simulations
  - 7.4 simulations/second
- Run a million simulations

Easy as 1…2…3…4.

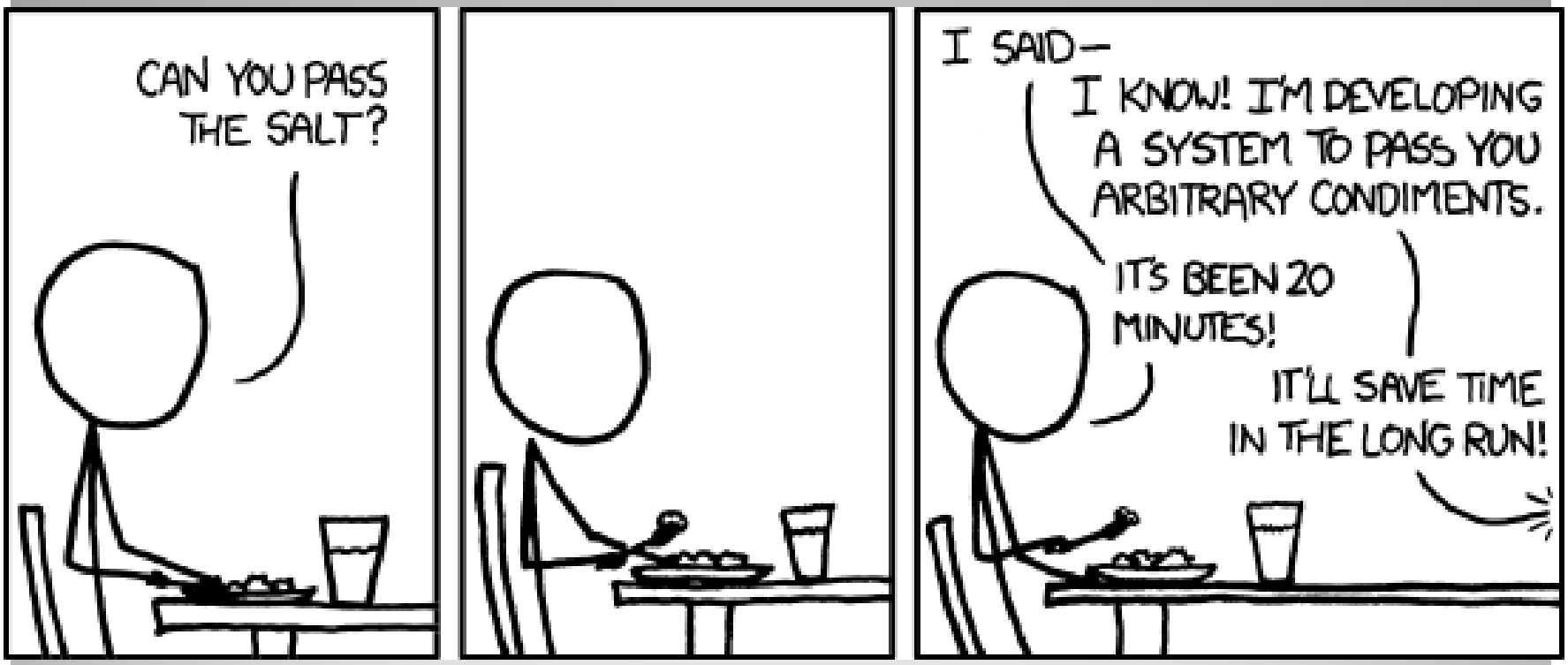


Results – Distance Travelled

Just a few small changes…

What, rev.2

- Given:
  - A fixed set of hardware components for a packet transmitter and receiver.
  - Video and telemetry streams that must go from the moon to Earth.
  - A stream of commands that must go from Earth to the rover on the moon.
  - Changing requirements from stakeholders.
- While(!Launched yet)
  - Create:
    - A finite state machine that provides the network and application layer control.

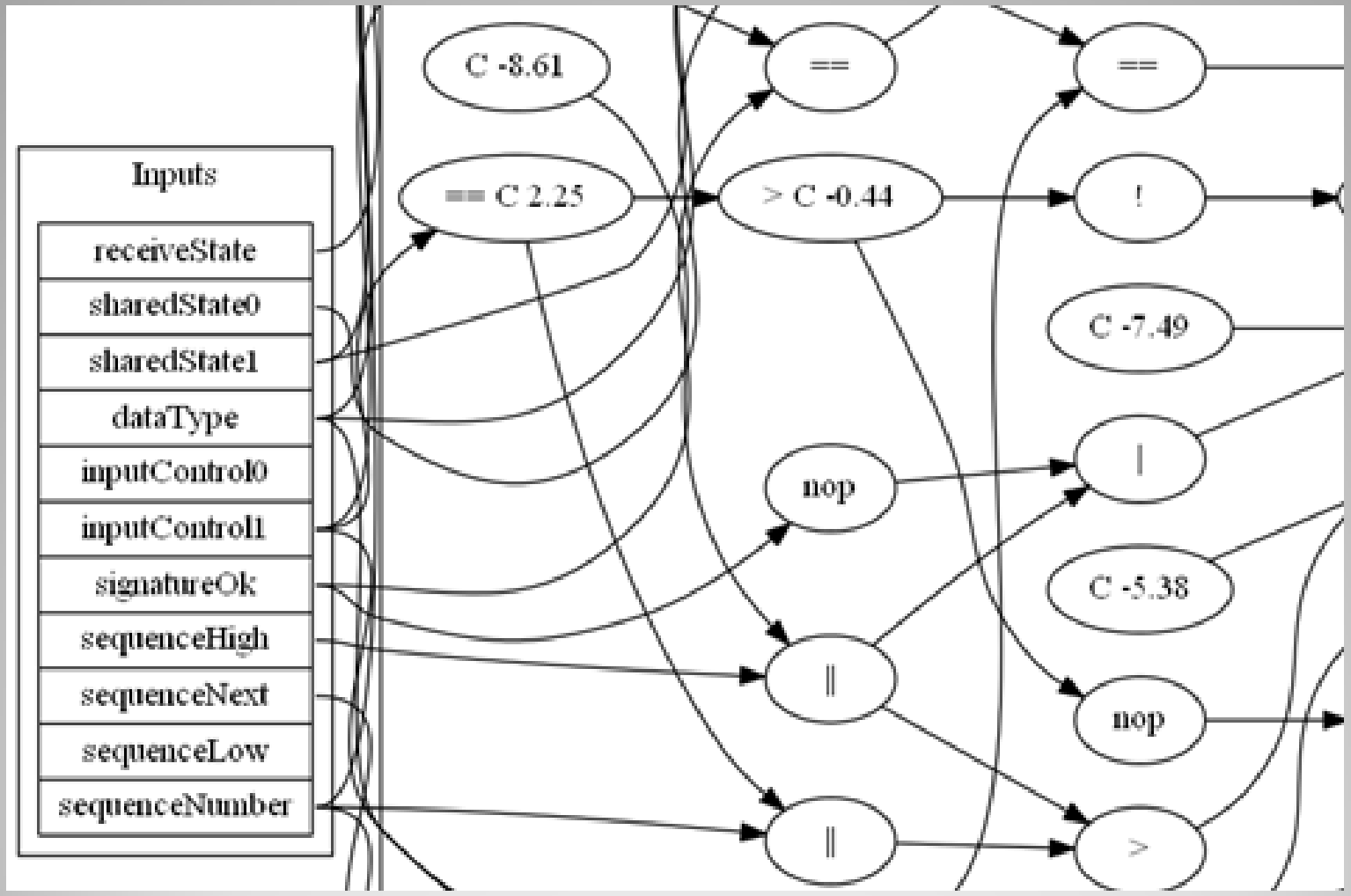


http://xkcd.com/974/
"The General Problem"

Can you pass the data?

Invention – GP

- Given:
  - A partial existing system structure
    - Inputs
    - Output(s)
    - ~~Formula structure~~
  - Test cases
  - Constraints
- Create:
  - ~~The optimal set of parameters~~
  - An equation or algorithm



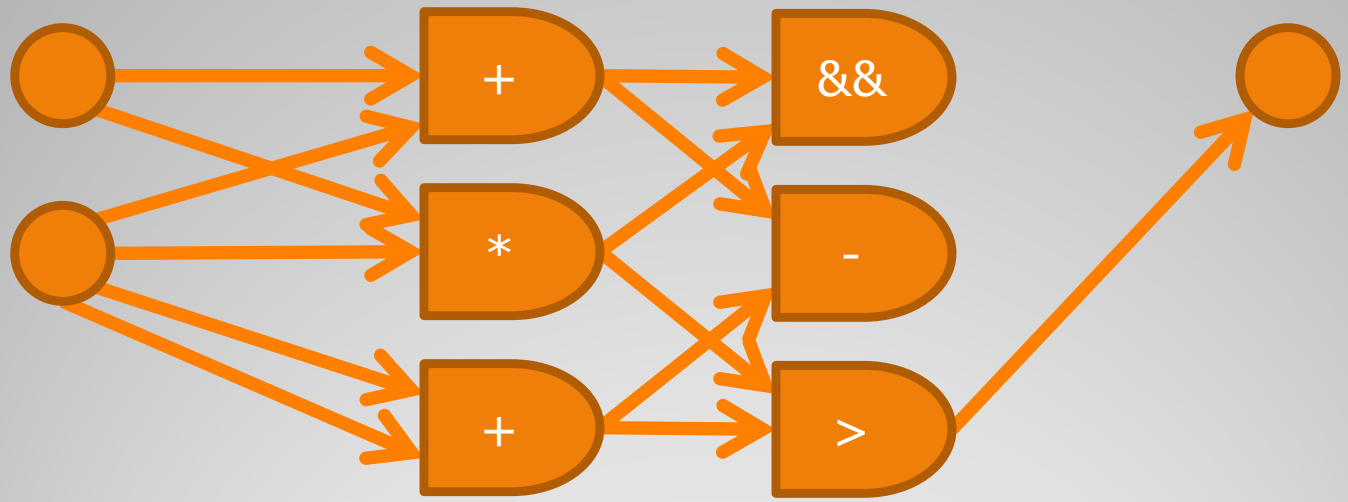
Invention - GP

How – CGP

- Cartesian Genetic Programming (CGP)
  - Generates equations like circuits.
  - Parallelizable results.
  - FPGA friendly.
  - Operators for simple math and logic.
    - Constant
    - Add, Subtract, Negative
    - Add Constant, Subtract Constant
    - NOP, !, &&, ||,
    - > Constant, >= Constant,== Constant
    - >, >=, ==

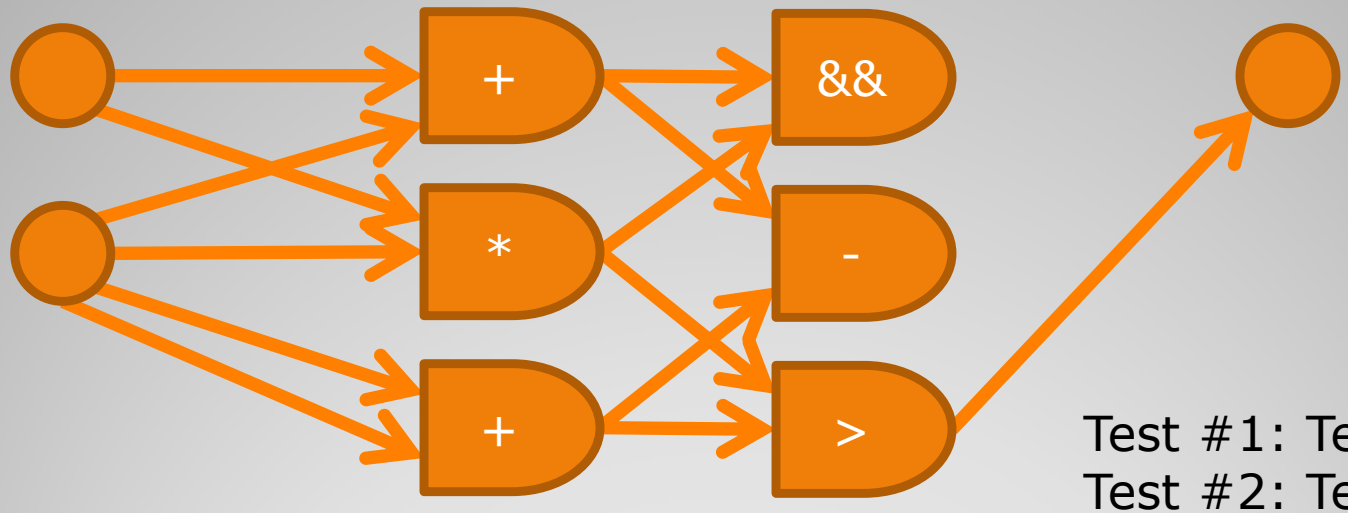
How – CGP

- Make a random “circuit”.



How – CGP

- Score the circuit.

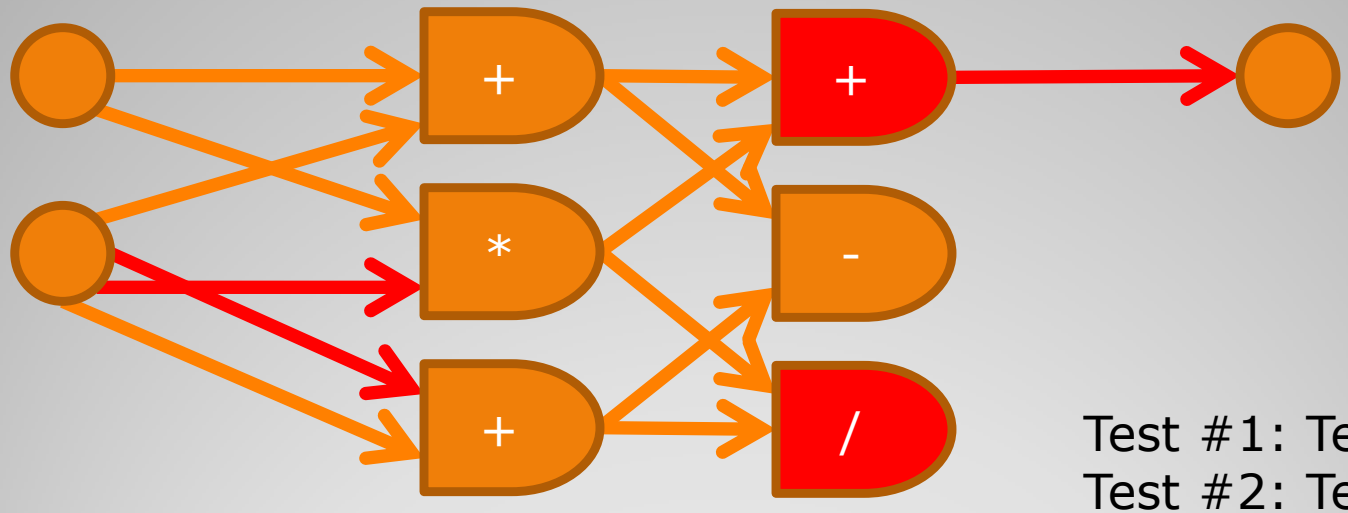


Test #1: Terrible
Test #2: Terrible
Test #3: Terrible

Score: Terrible*3

How – CGP

- Make random changes and rescore.



Test #1: Terrible
Test #2: Terrible-4
Test #3: Terrible

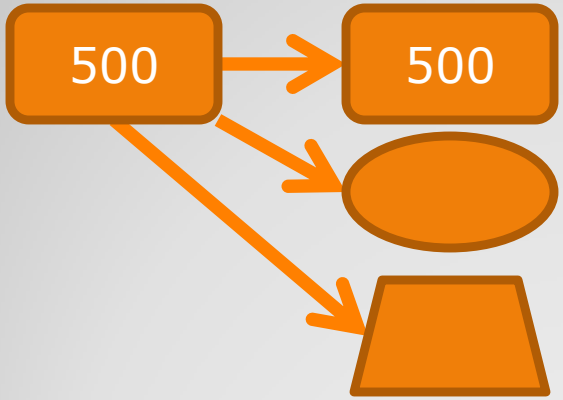
Score: Terrible*3-4

- Start with 1 parent.

500

How – CGP

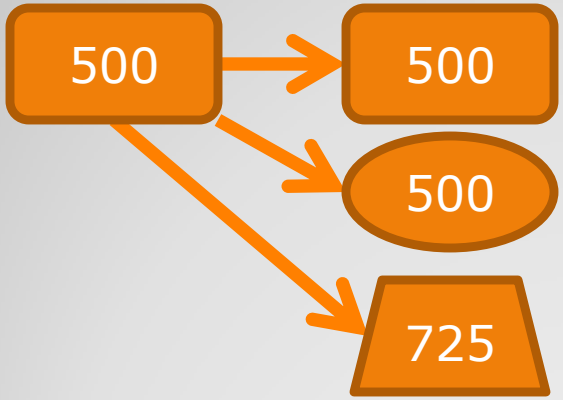
- Start with 1 parent.
- Make mutant children.



How – CGP

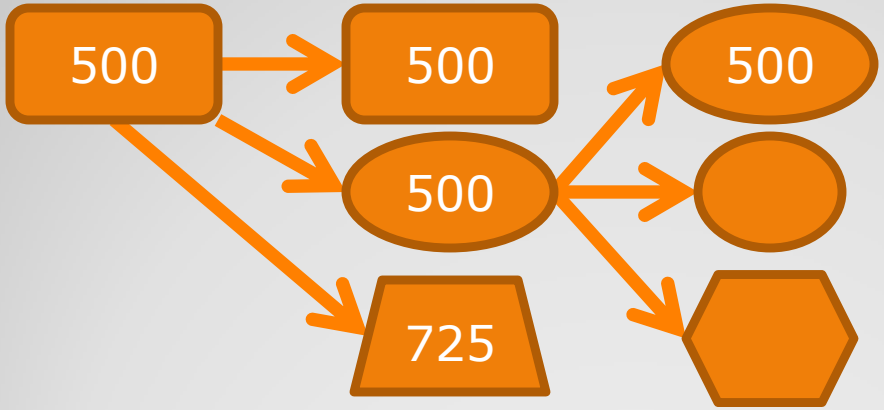
How – CGP

- Start with 1 parent.
- Make mutant children.
- Score everyone.



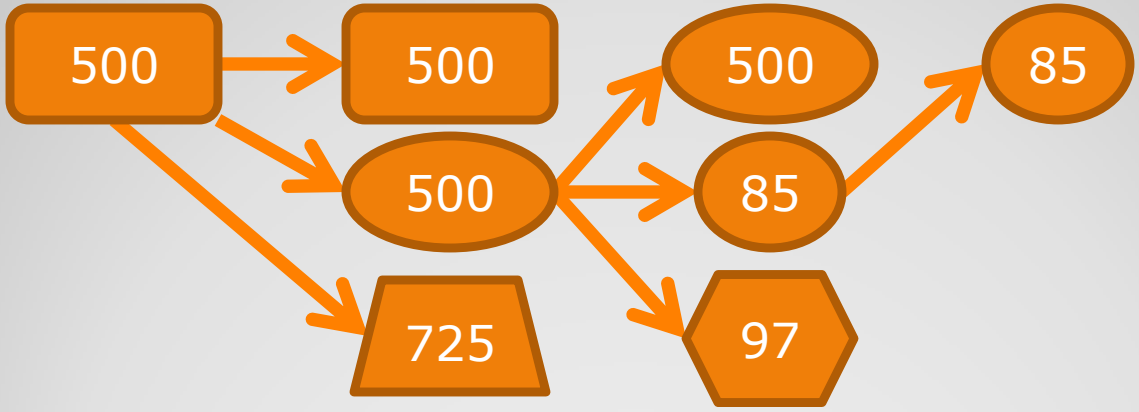
How – CGP

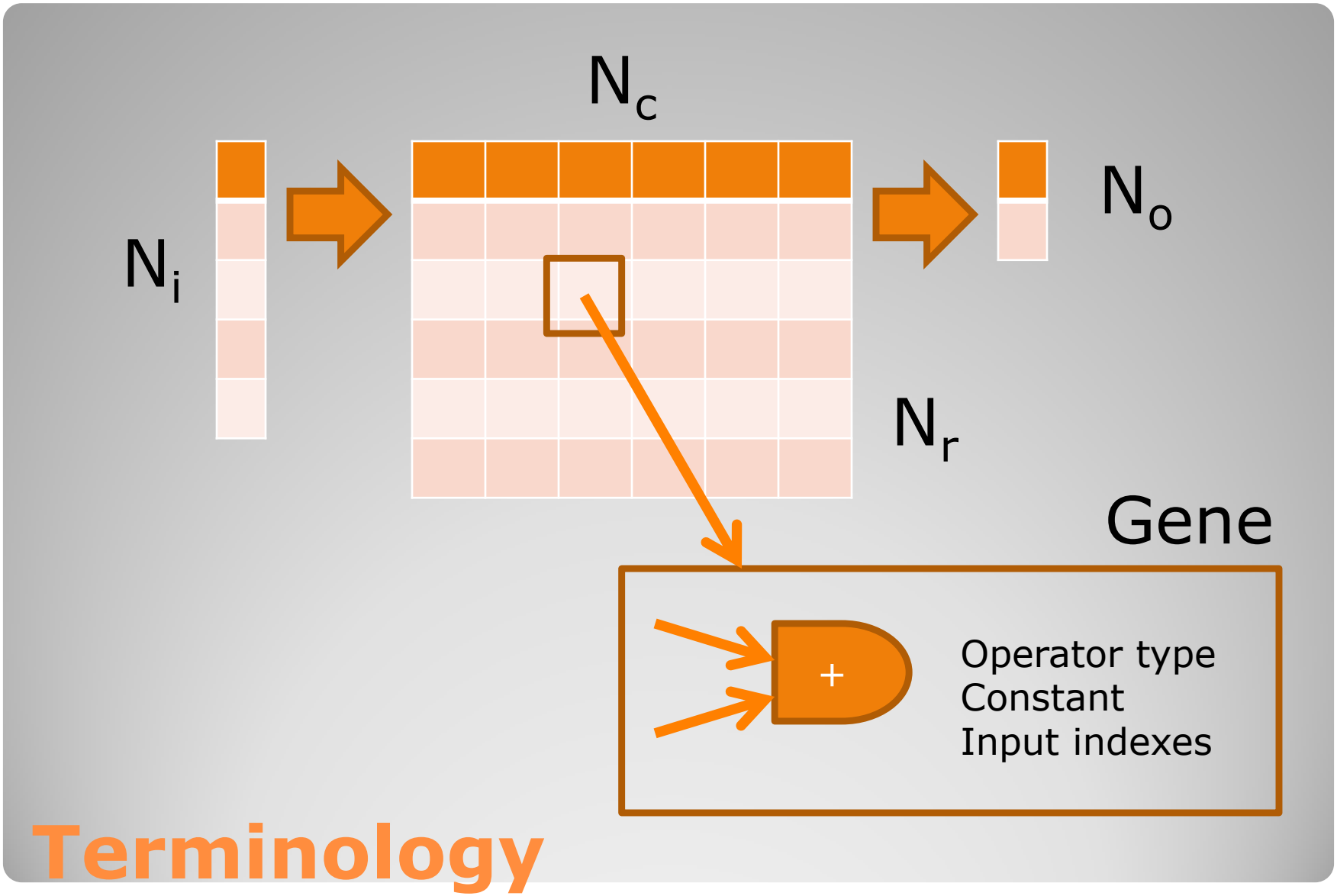
- Start with 1 parent.
- Make mutant children.
- Score everyone.
- Promote the best child that isn't worse than the parent.
  - Must promote anything equal to the parent!



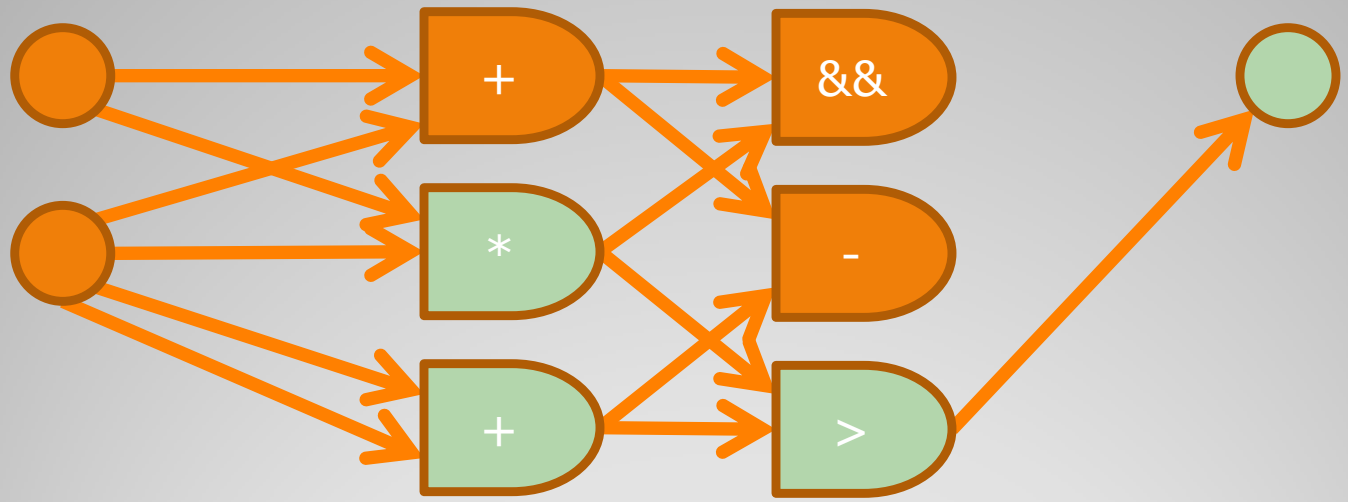
How – CGP

- Start with 1 parent.
- Make mutant children.
- Score everyone.
- Promote the best child that isn't worse than the parent.
  - Must promote anything equal to the parent!





Terminology



Runtime Optimization

- Individual
- Executor and Optimizer
- Mutator
- Population
- Support
  - Threadsafe, GPU-friendly random numbers
  - Save, Load
  - Images

Code – CGP Core

How – Attacker's Goals

- Mission Enders
  - Make rover execute attacker's commands
  - Prevent rover execution of real commands
  - Prevent acceptance of video data
- High Risk
  - Make mission control accept false telemetry
  - Delay rover execution of real commands

How – Simulation structure

- Discrete Event System simulation
  - Priority queue of events ordered by time
  - Set of Actors creating and handling events
- Packet is control flag array and payload
- Outer Space as random Actor
  - Lose
  - Replay
  - Echo
  - Corrupt

How – Simulation structure

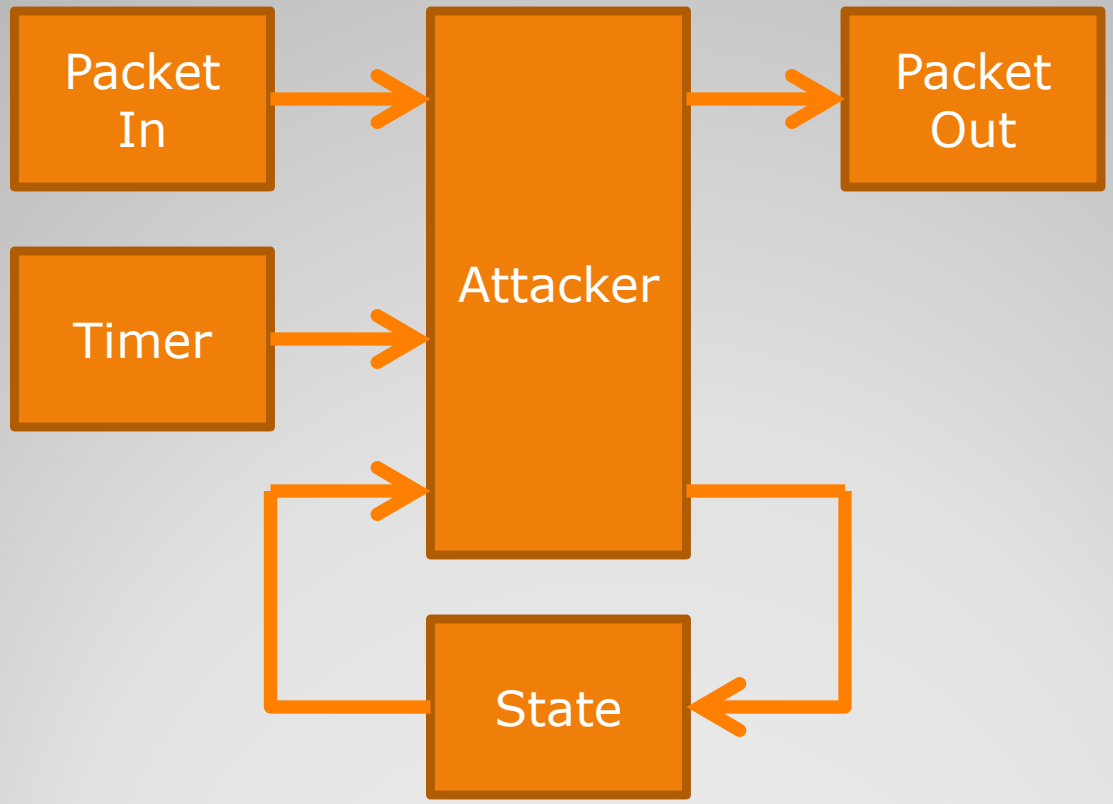
- Transceiver as set of Actors
  - Data Ready To Transmit
  - Data Received
  - Packet Expired in Transmit History
- Attacker as Actor

- Actors == Chromosomes
  - Scored separately
  - Promoted separately
  - Combines partial solutions for next generation

How – Attacker

- Coevolved
- Full knowledge of packet structure
- Can crack private keys
- Sees all packets
- Cannot prevent packet delivery
  - No “man in the middle” attacks
- Able to send to anyone
- Risk when transmitting

- Stateful and large for complex strategy
- Single Chromosome in feedback loop

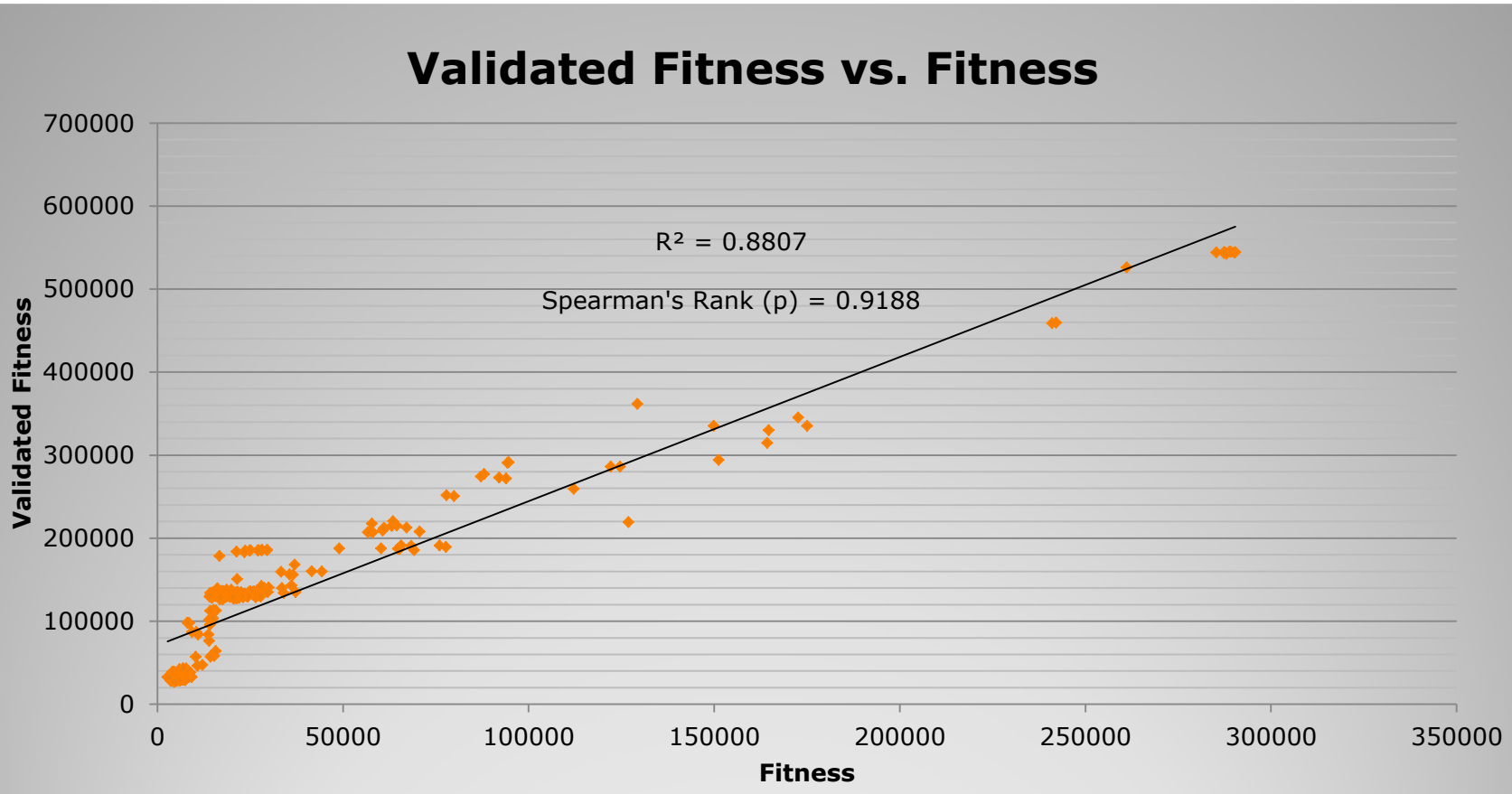


How – Attacker Structure

How – Fitness

- Generational fitness
- Best fitness
- Validated fitness

- Penalties/Rewards
  - Accepted bad signature
  - Accepted attacker packet
  - Accepted unknown data
  - Accepted data more than once
  - Never accepted data
  - Accepted data with wrong data type
  - Command packet accepted successfully



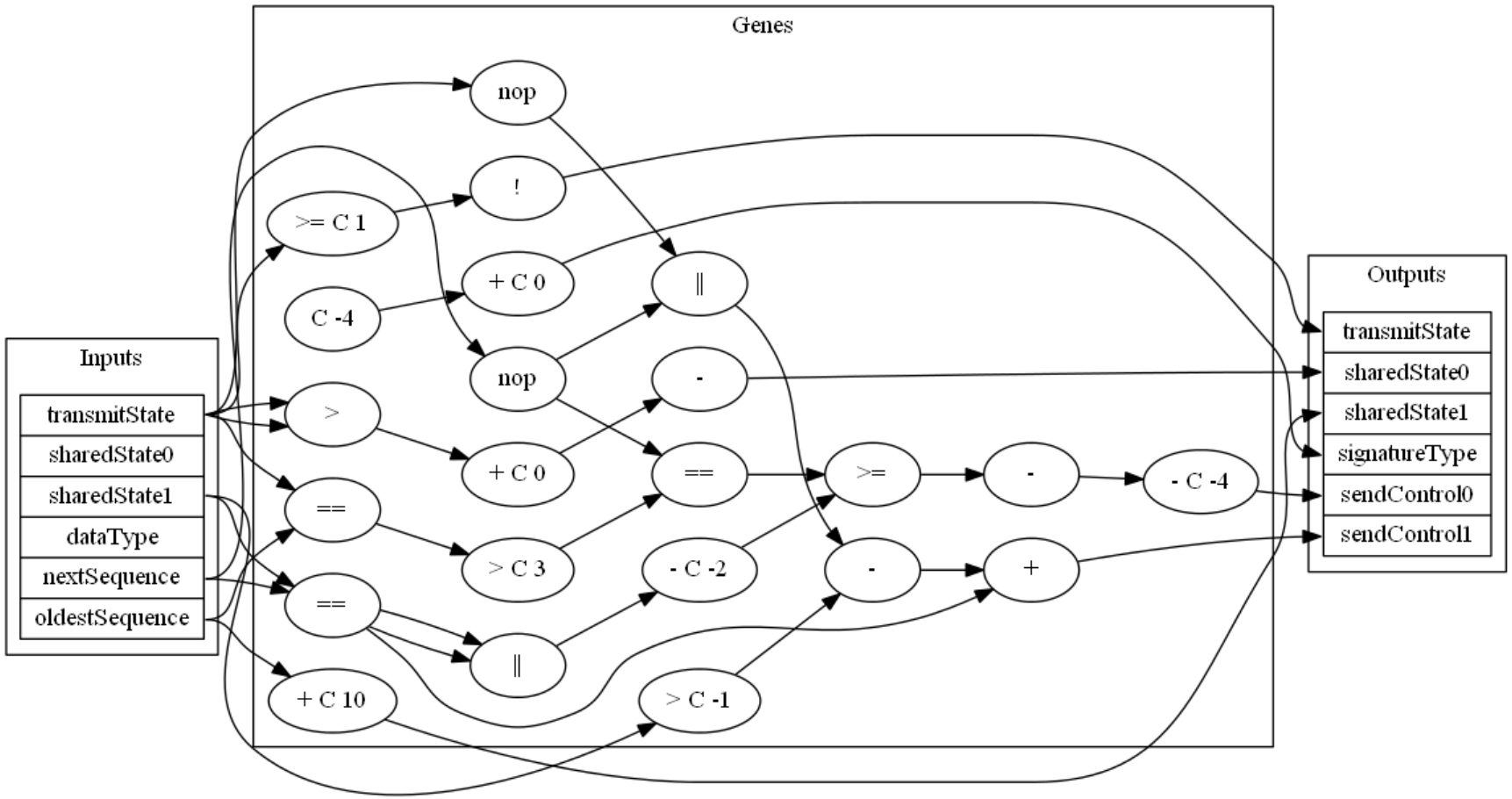
Tip – Validate Fitness

How - Questions to answer

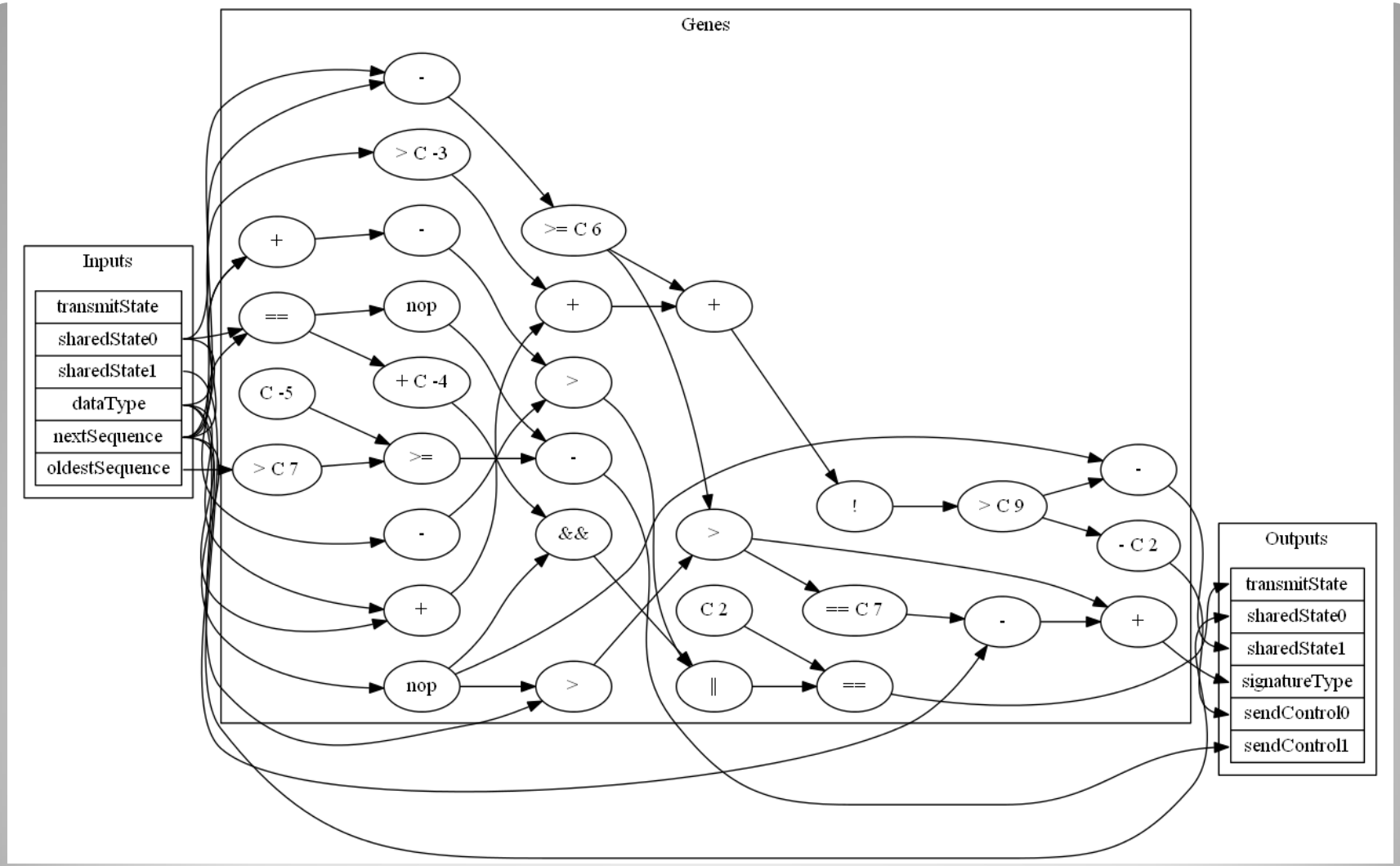
- Can the control logic be automatically created by Cartesian Genetic Programming (CGP)?
- If so, is the logic robust?
  - Can it work with poor signal quality?
  - Can it work with an attacker?
- Should the control logic be created:
  - In isolation?
  - Considering only poor signal quality cases?
  - Considering an attacker?
- Can an attacker's control logic be created at a faster pace than the network logic?

- Experiment
- Worlds
  - Data schedule
  - Actors
- Execution
- Fitness
  - Generational fitness
  - Best fitness
  - Validated fitness

Code – Simulation Core



Can CGP make control logic?



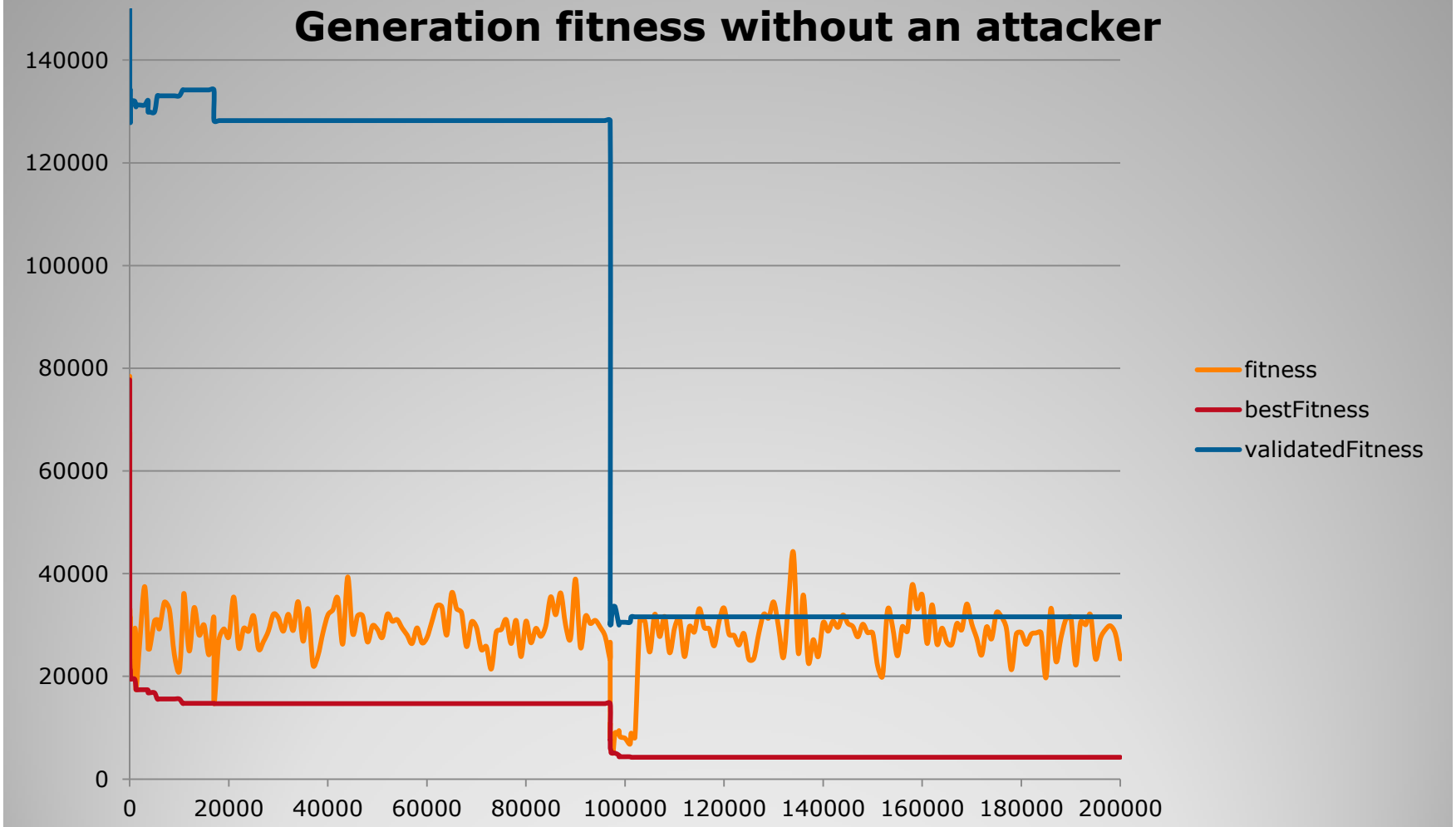
Can CGP make control logic?

More Tips

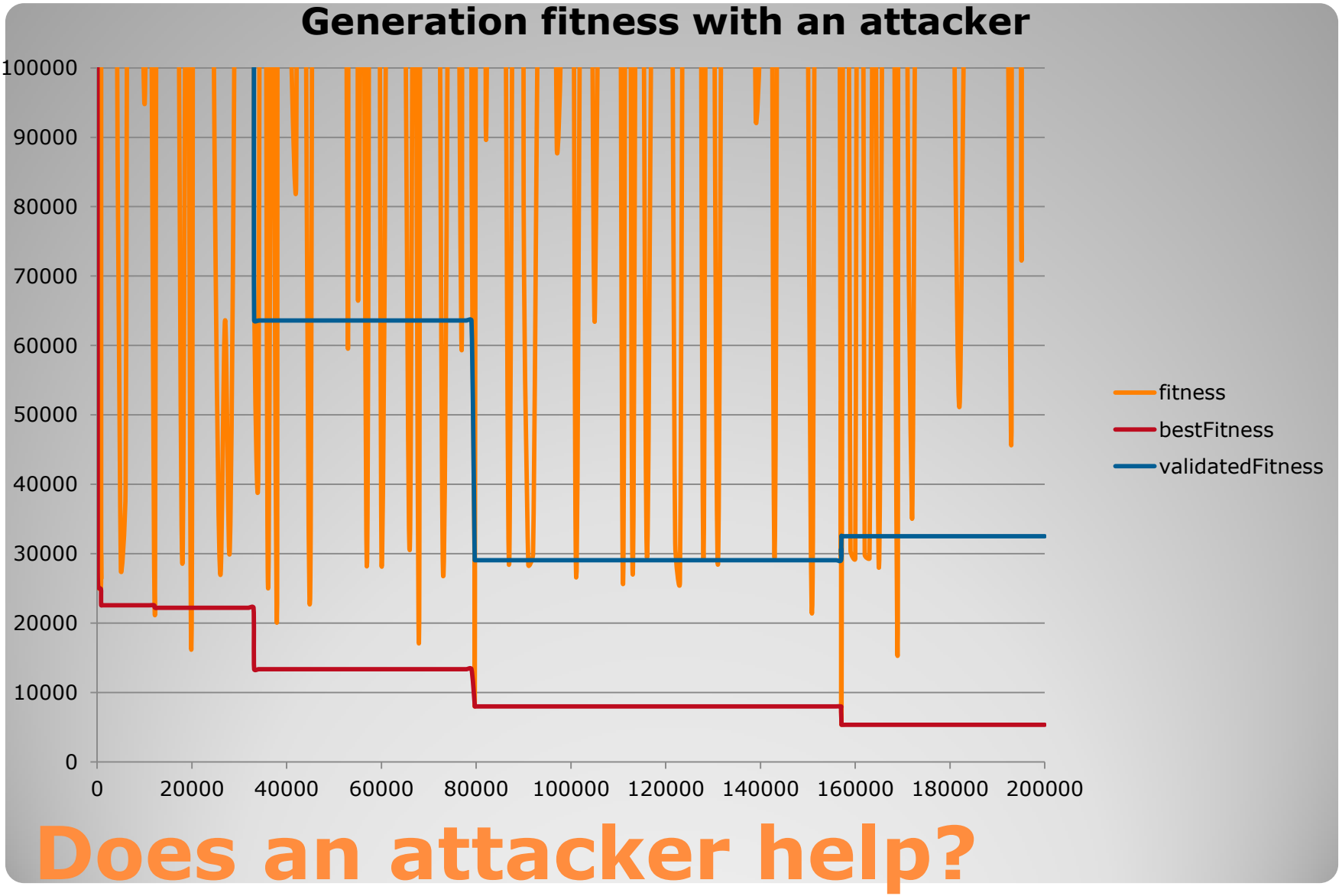
- Randomize everything!
- Reduce randomness in fitness
- Threadsafe random numbers
- Incentivize
  - “Mute” attackers
- Increase population size
- Dust-off your statistics books

Population Size Effects

	Population 4	Population 8
Fitness Evaluations		2x
Run time		3x
Same generations		Much better fitness
Same run time	Equal fitness	Equal fitness



Does an attacker help?



Does an attacker help?

- Does an attacker's presence produce a better transceiver?

	No attacker	With attacker
Best fitness	18316	23208
Validated fitness	134542	137494
Active genes		+50%
Fitness:Generation correlation	Medium	Near Zero
Other Fitness:Generation	Medium	Medium
Val. Fitness:Gen. Slope		2x

Does an attacker help?

Results - Questions answered

- Can the control logic be automatically created by Cartesian Genetic Programming (CGP)?
- If so, is the logic robust?
  - Can it work with poor signal quality?
  - Can it work with an attacker?
- Should the control logic be created:
  - In isolation?
  - Considering only poor signal quality cases?
  - Considering an attacker?
- Can an attacker's control logic be created at a faster pace than the network logic?

Future

- CGP on GPU
- Worlds on GPU
- Finer-grain Chromosomes
- New “best” selection with an attacker
- Islands
- Freezing transceiver or attacker
- Attacker detection and countermeasures

- Danke!
- <shameless recruiting ad>
  - Join us at ptscientists.com!
- </shameless recruiting ad>
- Wes Faler of Part-Time Scientists
  - wf@ptscientists.com

</presentation>

Resources

- Code and presentation
  - http://wp.me/pGgFl-V
- Julian Miller (inventor of CGP)
  - http://sites.google.com/site/julianfrancismiller/professional
- "Cartesian Genetic Programming" book
  - http://www.springer.com/computer/theoretical+computer+science/book/978-3-642-17309-7
- "Evolved to Win" e-book
  - http://www.moshesipper.com/etw/
- "Communication Protocol Engineering" book by Miroslav Popovic