

Return Oriented Program Evolution with ROPER

May 19, 2020

Contents

Introductory Remarks

The Concept of Return-Oriented Programming

The Fundamental Problem of Cybersecurity

At bottom, there is no essential distinction between data and code.

"Data" is just information your system trusts.

The Stack

Top of Calling Stack Frame

2nd argument

1st argument

Return Address

Locally Scoped
Data, including
Stack-allocated
Variables

- the hacker feeds some input data to the process
- which is written to a buffer in stack memory
- but which overruns the buffer
- corrupting the frame's return address

The Stack, Smashed

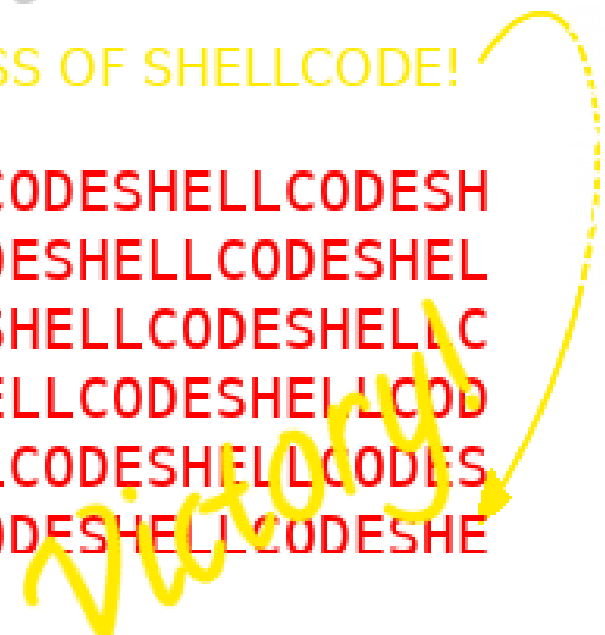
Top of Calling Stack Frame

2nd argument

1st argument

ADDRESS OF SHELLCODE!

SHELLCODESHELLCODESH
ELLCODESHELLCODESHEL
LCODESHELLCODESHELLC
ODESHELLCODESHELLCOD
ESHELLCODESHELLCODES
HELLCODESHELLCODESHE



- so that it points into the buffer
- a buffer that turns out to contain machine code
- to which the program counter "returns"
- executing it just as it would its own instructions!

DEP / $W \oplus X$

Top of Calling Stack Frame

2nd argument

1st argument

ADDRESS OF STACK DATA...

HARMLESSDATAHARMLESS
DATAHARMLESSDATAHARM
LESSDATAHARMLESSDATA
HARMLESSDATAHARMLESS
DATAHARMLESSDATAHARM
LESSDATAHARMLESSDATA

segfault!

- One way of mitigating this is to try to ensure that no page of memory is both writeable **and** executable.
- The idea being that *data* should be writeable, but never executable, while *code* should be executed, but not written at runtime.

Subverting $W \oplus X$

- $W \oplus X$ may prevent the *execution* of input data, but it doesn't prevent attempts to *return* to that data.
- Why should the hacker need to supply their own machine code?
- There's quite a bit just laying around, in executable memory.
- Why not just build a payload with whatever's handy?

$W \oplus X$ is a Leaky Abstraction

- It rests on all-too-narrow concepts of "instruction" and "execution".
- The payload's *instructions* don't need to be bytes of machine code.
- They just need to influence control flow, in a controllable way.

So is the *Structured Programming Machine Model*

- The machine model on which structured programming is based already carves up an executable into chunks that **return** control after being dispatched.
- To the programmer, these are "functions", but this is too granular a viewpoint.
- *Any* chunk of code ending with a **return** returns control to whomever controls the stack.
- And our data controls the stack!

The ROVM supervenes on the SPMM



- Chunks of code that return control are called "gadgets".
- They form a spontaneous ISA, whose **program counter** is the **stack pointer** of the underlying architecture.
- Let's call this ISA a "Return-Oriented Virtual Machine".

We can program this machine with input data

- All we need to do is to discover and supply a buffer of instructions.
- These are not instructions for the underlying architecture, but for the ROVM.
- $W \oplus X$ is blissfully unaware of the ROVM, and powerless to prevent us from executing data as *ROVM* code.

Genetic Programming

TODO Genetic Algorithms

- Variation (mutation and crossover)
- Selection (fitness function)
- Reproduction (iteration)

Genotype \rightarrow Phenotype

- Genetic programming turns on an analogy between genotype \rightarrow phenotype maps on the one hand, and, on the other, the relation between a program's syntax and its operational semantics.
- The syntactical representation of a program is its genotype, and its semantic behaviour is its phenotype.

Exploring Weird Machines through Genetic Programming

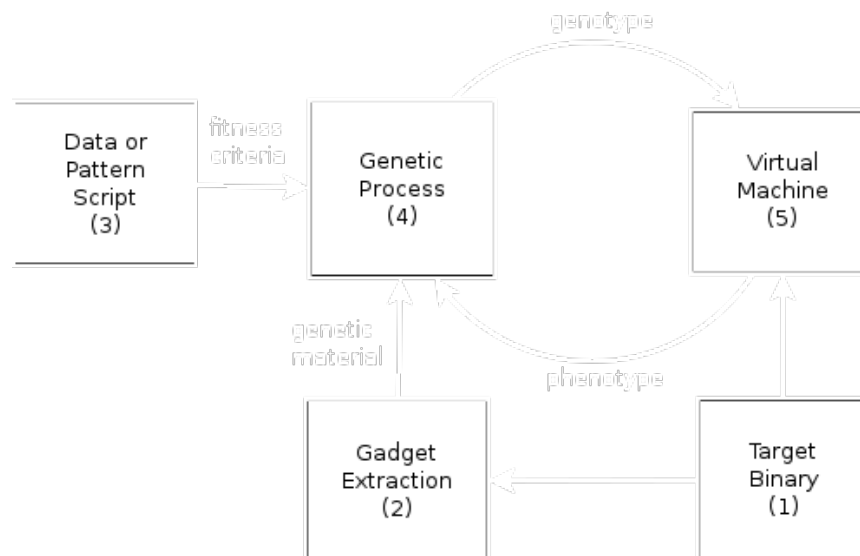
- We are often going, blind, into terra incognita.
- Evolutionary computation has shown surprising creativity in discovering and exploiting computational environments. (See The Surprising Creativity of Digital Evolution for examples.)
- The irregular, side-effect-rich character of the computational primitives exposed by many WMs, ROP included, make them difficult for humans to reason about.

Challenges that ROP exploration poses for GP

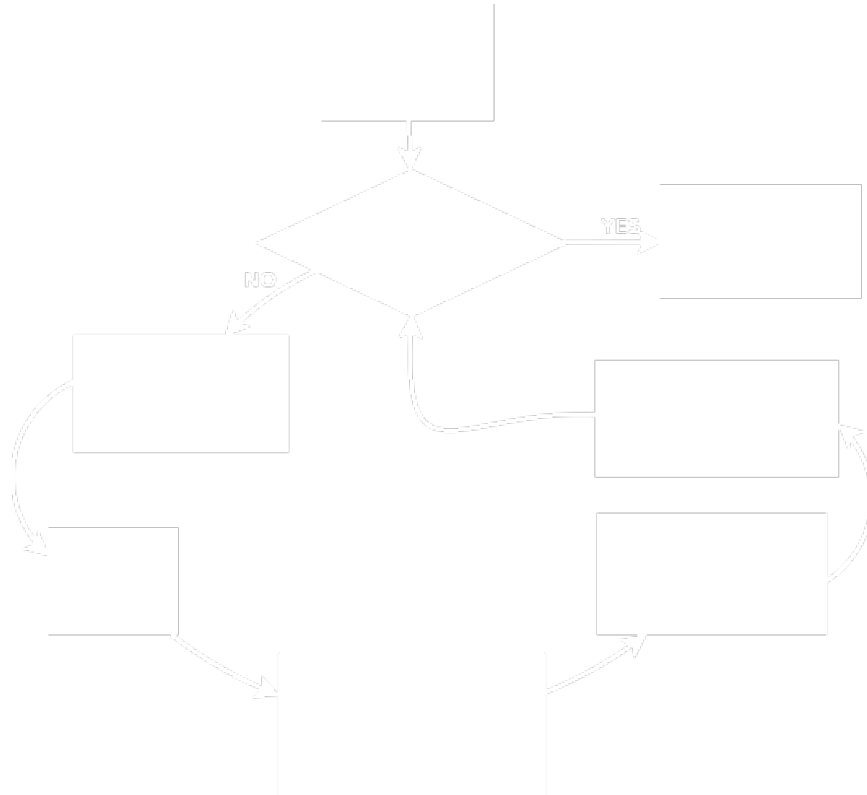
- GP typically employs highly specialized and parsimonious virtual machines, tailored to the problem set in question.
- Our "instruction set" is the set of "gadgets" we happen to discover in a binary.
- This set is not small (often numbering in the hundreds, or more).
- Nor "tailor made".
- Nor is it evenly distributed over the semantic space it represents.

Design and Implementation of ROPER

Bird's eye view



Tournament Selection



Genomic Structure

- Each genome is a one-dimensional *chain* composed of *clumps*.
- A *clump* is a gadget address a , followed by $SP_{\Delta}(a) - 1$ machine words
- where $SP_{\Delta}(a)$ is the (estimated) number of words that $*a$ will pop from the stack, when run.
- Several "epigenetic" fields of metadata are also associated with both the *chain* and *clump* structures.

Genetic Operators: Clumpwise Mutation

- address substitution
- arithmetical & logical manipulation of dwords

- indirection/dereference of dwords
- permutation of pairs of dwords

Genetic Operators: Chainwise Crossover

- restricted to single-point crossover
- splice point selected by weighted random choice, using the average of each link's previous hosts' fitness scores, to favour adaptive gene linkage
- recently, a mechanism to promote homologous crossover in fitter specimens has been introduced

Experimental Studies

Tasks and Fitness Functions

- An arbitrary and inscrutable fitness function
- System call preparation
- Classification tasks:
 - An artificial, linearly-separable dataset
 - The Iris dataset
- A Snake game

System Call Preparation

- The goal here is to prepare the CPU for a system call, with the registers containing and pointing to the necessary arguments.
- The fitness function uses a combination of numerical distance and bit-wise hamming distance, for immediate values, and memory proximity for indirect values.
- A successful evolutionary run delivers a payload that can be used for practical purposes.

System Call Preparation

Champion of the *Wiwzuh* population:

```
0000b4ac      pop {r4, r5, r6, r7, r8, pc}

0000d1a0      cmp r0, #0
0000d1a4      popeq {r3, r4, r5, pc}

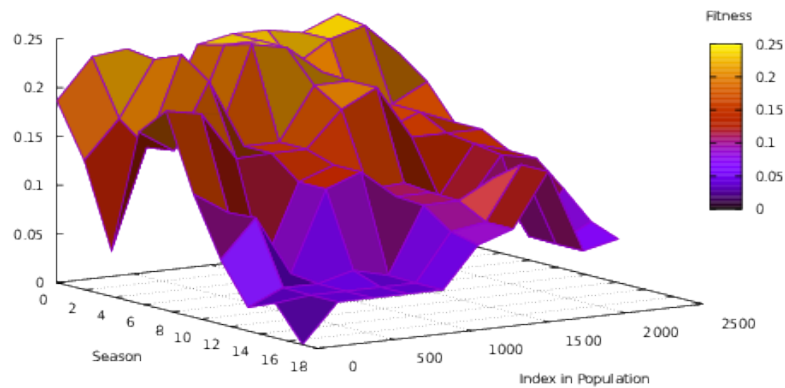
00016654      cmp r0, #0
00016658      ldr r3, [pc, #4]
0001665c      moveq r0, r3
00016660      pop {r3, pc}

0001706c      ldm sp, {r0, r1}
00017070      add sp, sp, #0x10
00017074      pop {r4, r5, r6, pc}

;; R0: 0001f62f  R2: 00000000
;; R1: &0001f62f  R7: 0000000b

;; to call execv("/tmp/flashXXXXXX", ["/tmp/flashXXXXXX"], NULL)
00018fc4      svcvc #0xffffffff
```

Historical Profile of the *Wiwzuh* population



The Enigma of Stray Gadgets

- This task also produced a number of specimens whose traces are too long and complex to display in detail here, but which were especially interesting for their labyrinthine nature, and the degree to which their execution traces strayed from the harvested gadget set.
- I will nevertheless **try** to display one here.

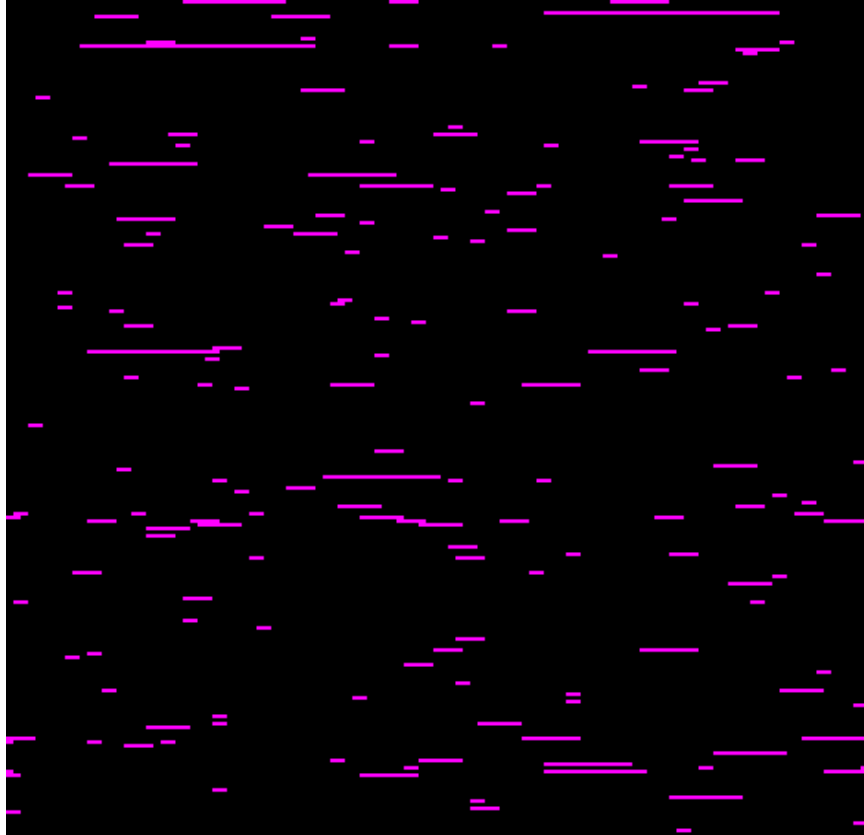
The Enigma of Stray Gadgets

The Enigma of Stray Gadgets

These were of interest in two respects:

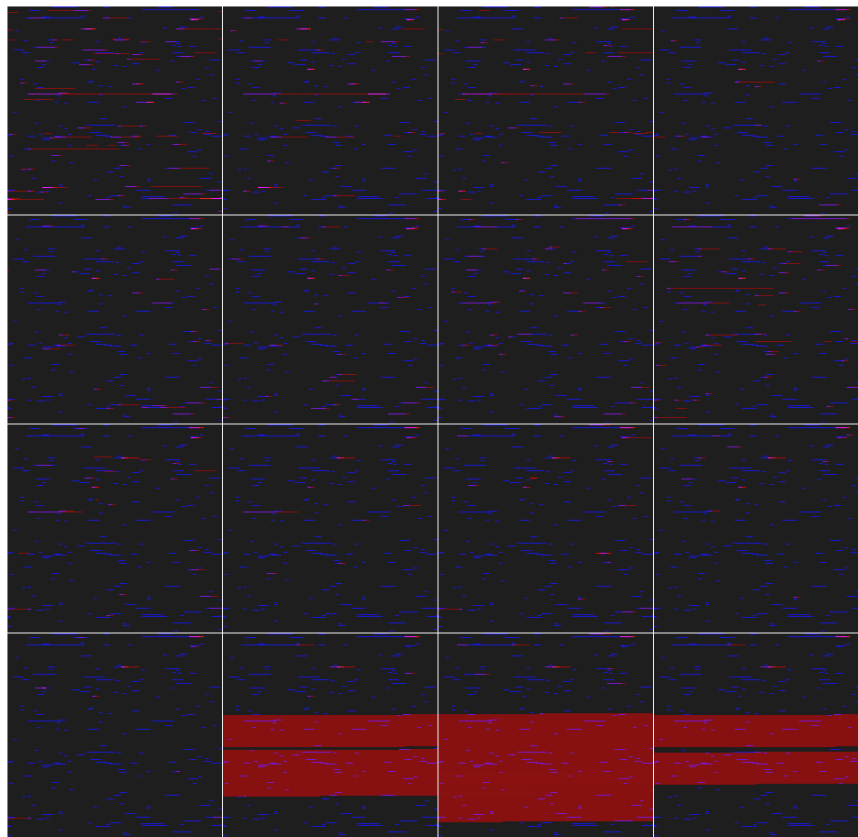
- they contained complex *heuristic breakers* making them likely to bypass various IDS systems in the literature, as a sheer evolutionary *spandrel*
- theoretically, their behaviour was enigmatic. Straying is dangerous for chains, and comes with great risk of crashing, yet it appeared with *prima facie* improbable frequency in our populations.

The Environment



Distribution of gadgets in `tomato-RT-N18U-httpd`.

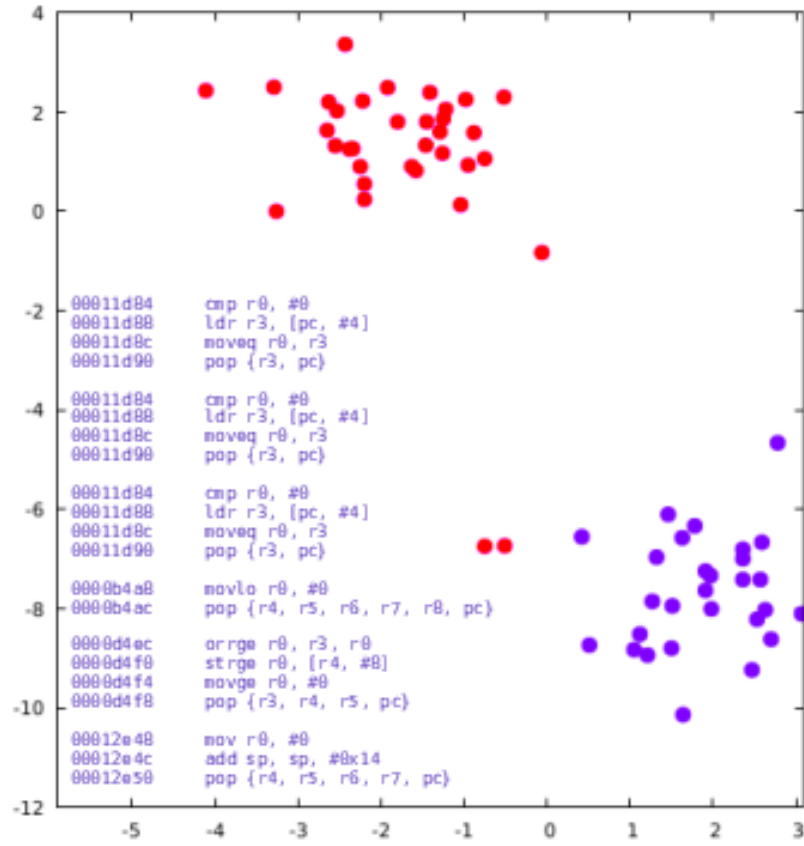
The Use of the Environment



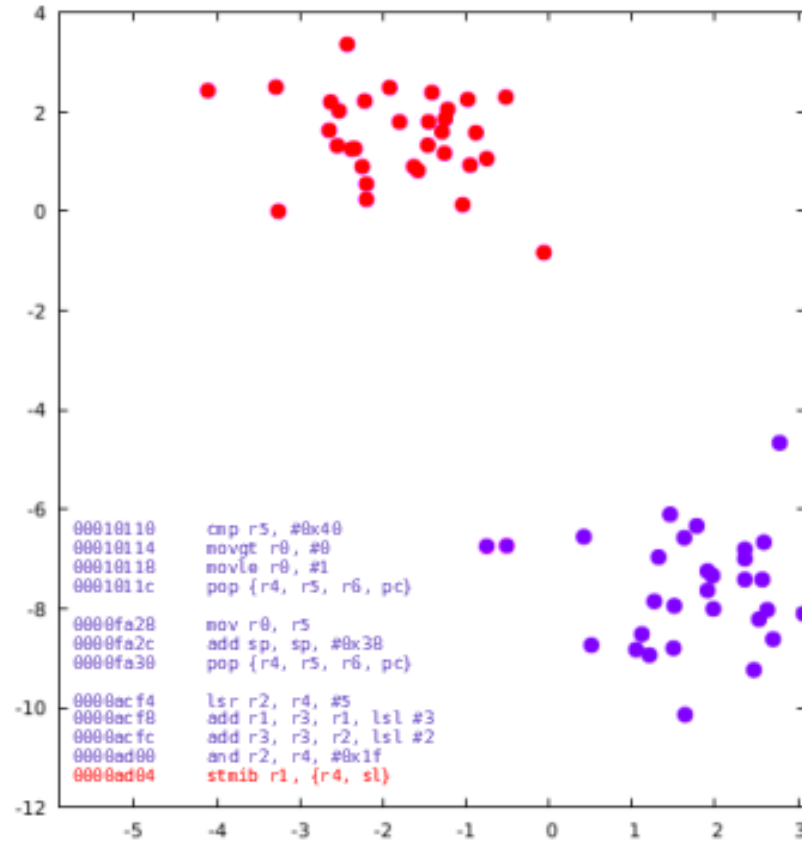
A simple classification task

- For the classification tasks, I initially used a common, bid-based algorithm to map behaviour to classification decisions on data samples.
- A set of output registers was mapped to the class list, and data was classified according to the register containing the greatest signed value.

Fair initial results

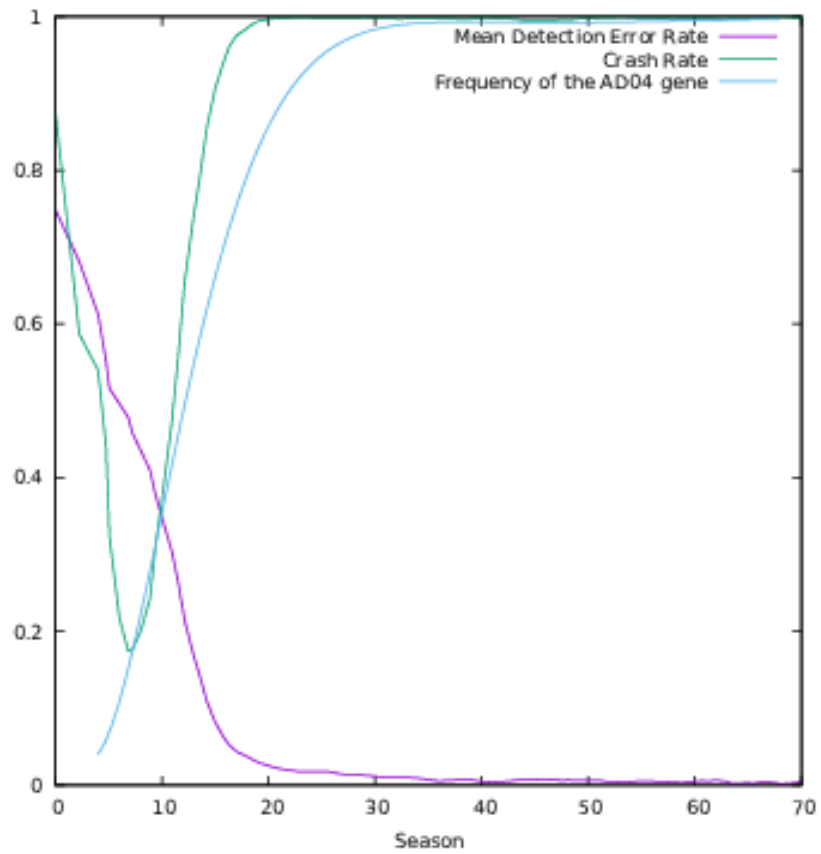


An interesting case of malignancy

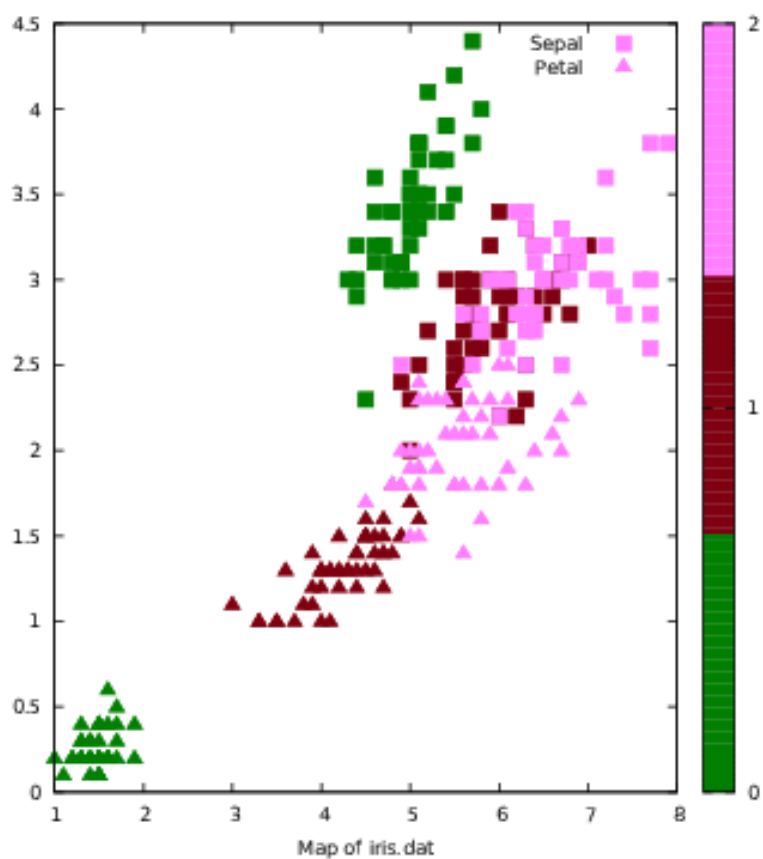


Here, the gene responsible for correct classification of the data was also responsible for crashing the execution. It rapidly took over the population.

An interesting case of malignancy



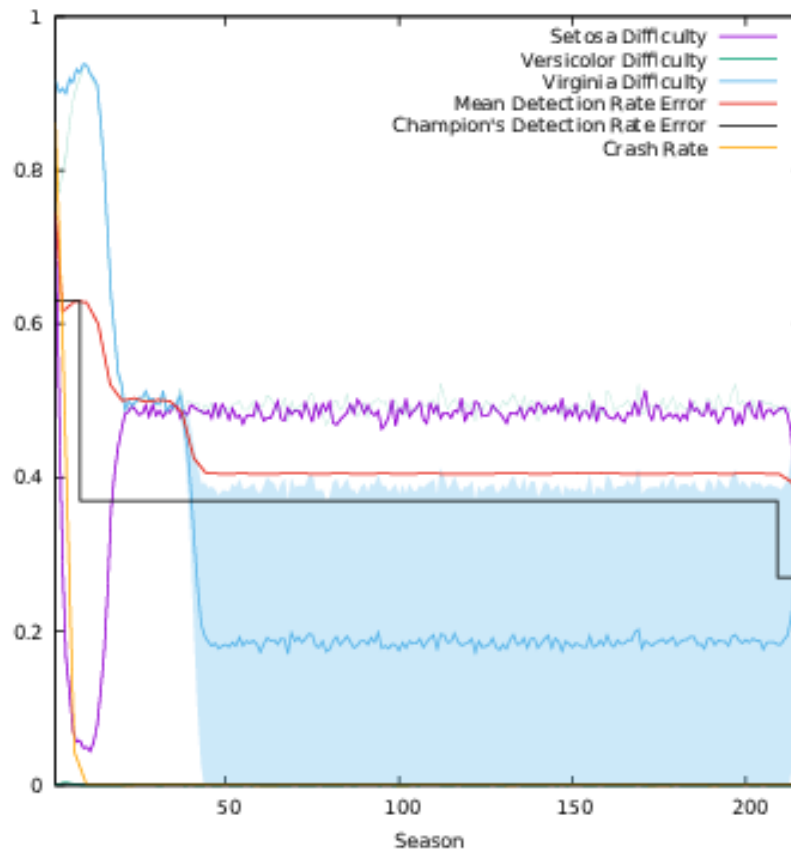
The Iris Dataset



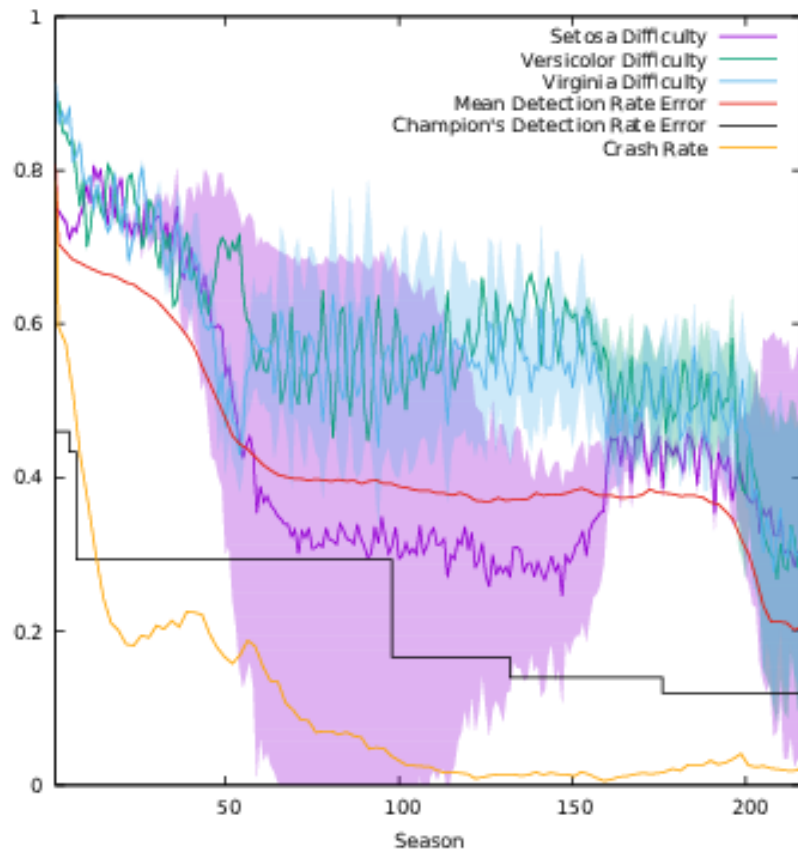
ROPER on the Iris Dataset

- This dataset proved a serious challenge for ROPER, which rarely achieved better than a 66% detection rate (using the bid-bin method).
- Success only came with the introduction of a fitness sharing mechanism.

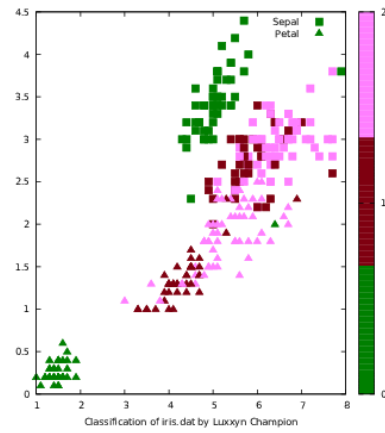
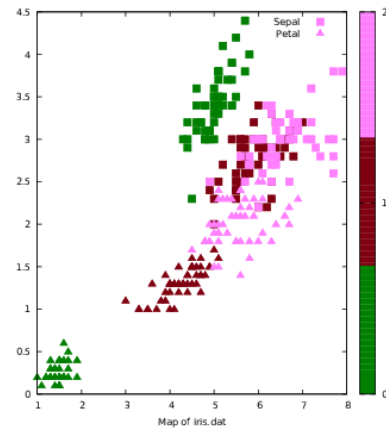
Iris without Fitness Sharing



Iris with Fitness Sharing



Iris as Classified by ROPER



Questions?