Detection of Web Defacements by means of Genetic Programming

Eric Medvet, **Cyril Fillon** and **Alberto Bartoli**

DEEI, University of Trieste, Italy

Web site defacement

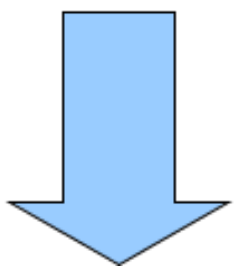
- **Common** web attack: consists in replacing part or entire web page…
  - …with **evident** disturbing content or political, social, religious messages
  - …with **subtle** changes (links, forms, …)
- More than **490,000** pages defaced in 2006, about 1500 pages/day

Detection: requirements

- Monitoring dynamic web pages **automatically**, that is:
- …without any assumption on expected content, appearance, behavior of the page

Detection: key idea

1. Observe the monitored page
2. Build a **profile**
3. Detect **deviation** from the profile

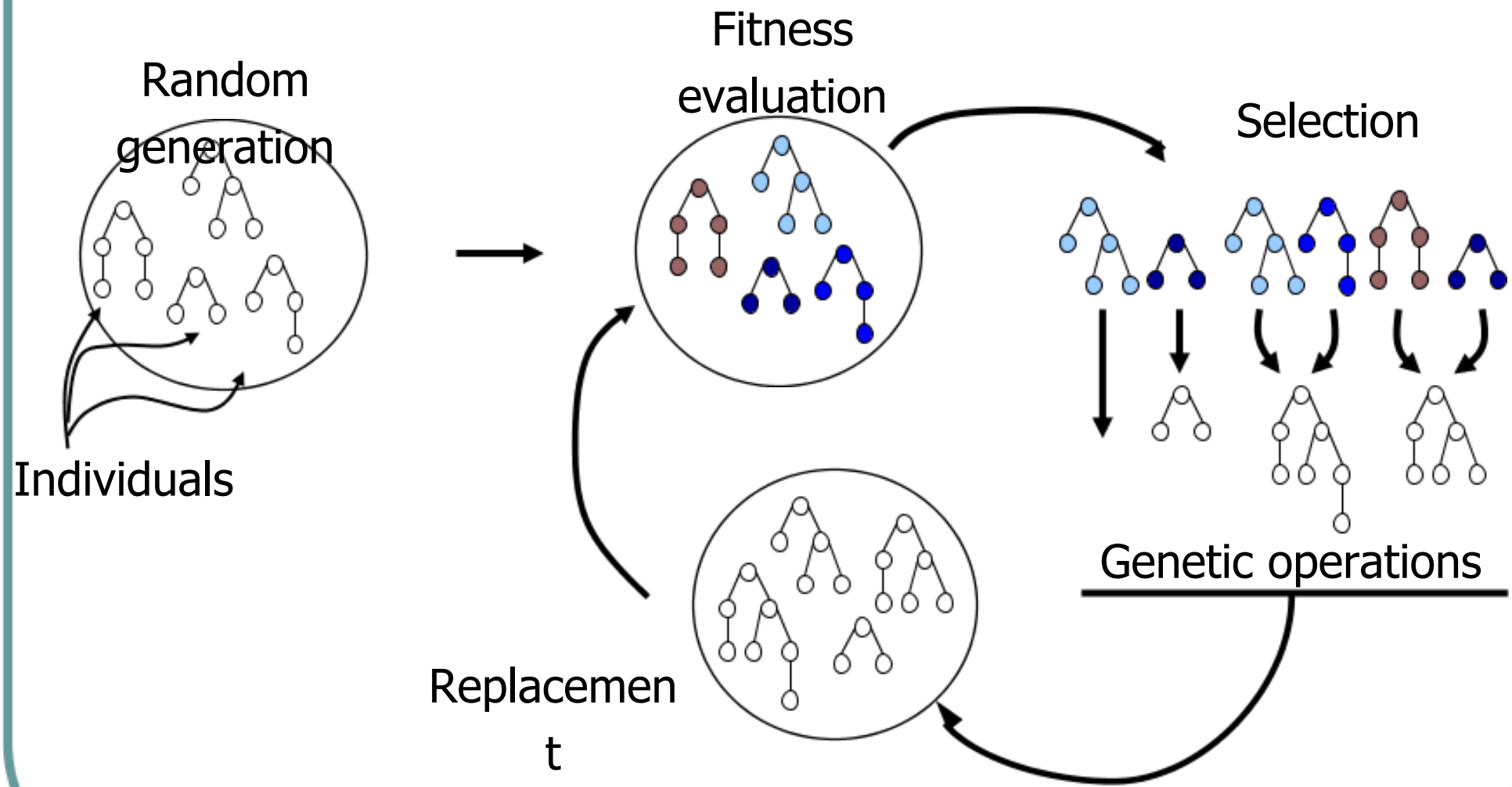


- This work approach: using **Genetic Programming (GP)** for points 2 and 3

Genetic Programming overview (I)

- The process of **solving** a problem by **searching** in a space of possible computer programs for the **fittest** individual computer program

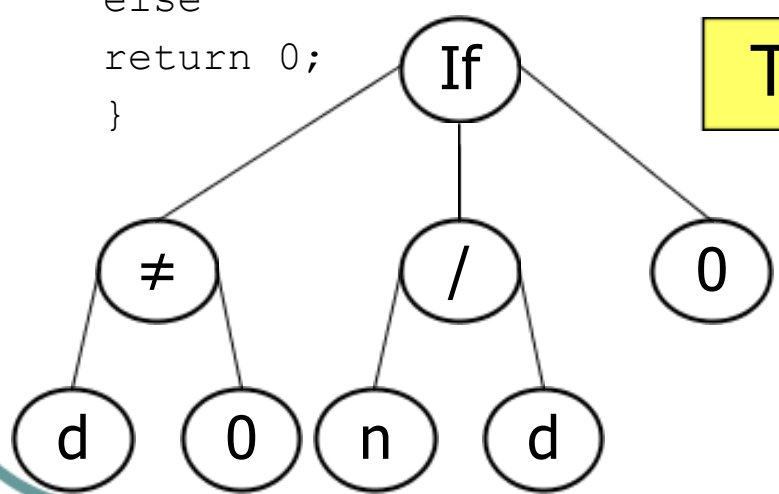
Genetic Programming overview (II)



GP overview: individual

- Each individual is a parse tree representing a **program** or **mathematical expression**

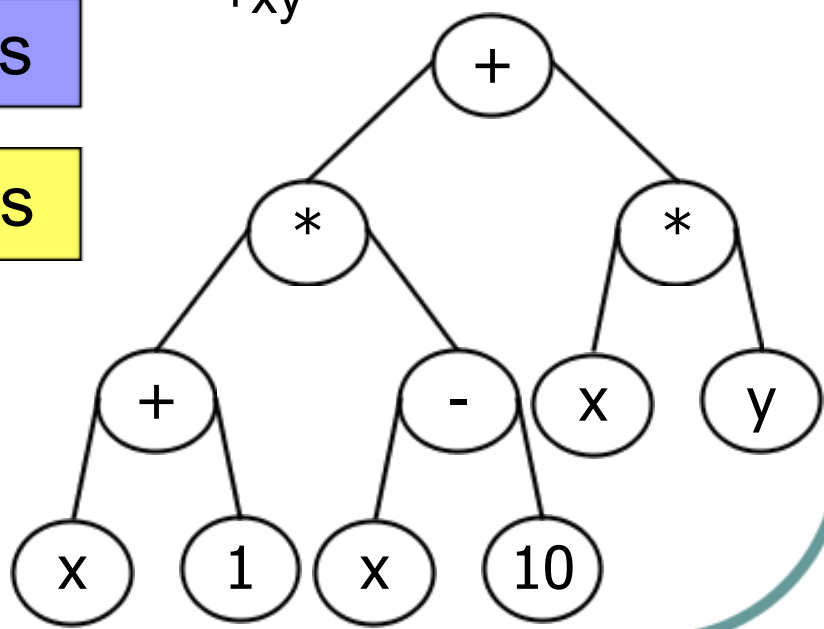
```
double doSomething(n, d)
{
if (d!=0)
return n/d;
else
return 0;
}
```



Functions

Terminals

F(x,y) = (x+1)(x-10)
+xy



GP overview: main params

1. Functions and terminals
2. Fitness function
3. Stop criterion for iterative process

- Chosen basing on the problem knowledge and domain

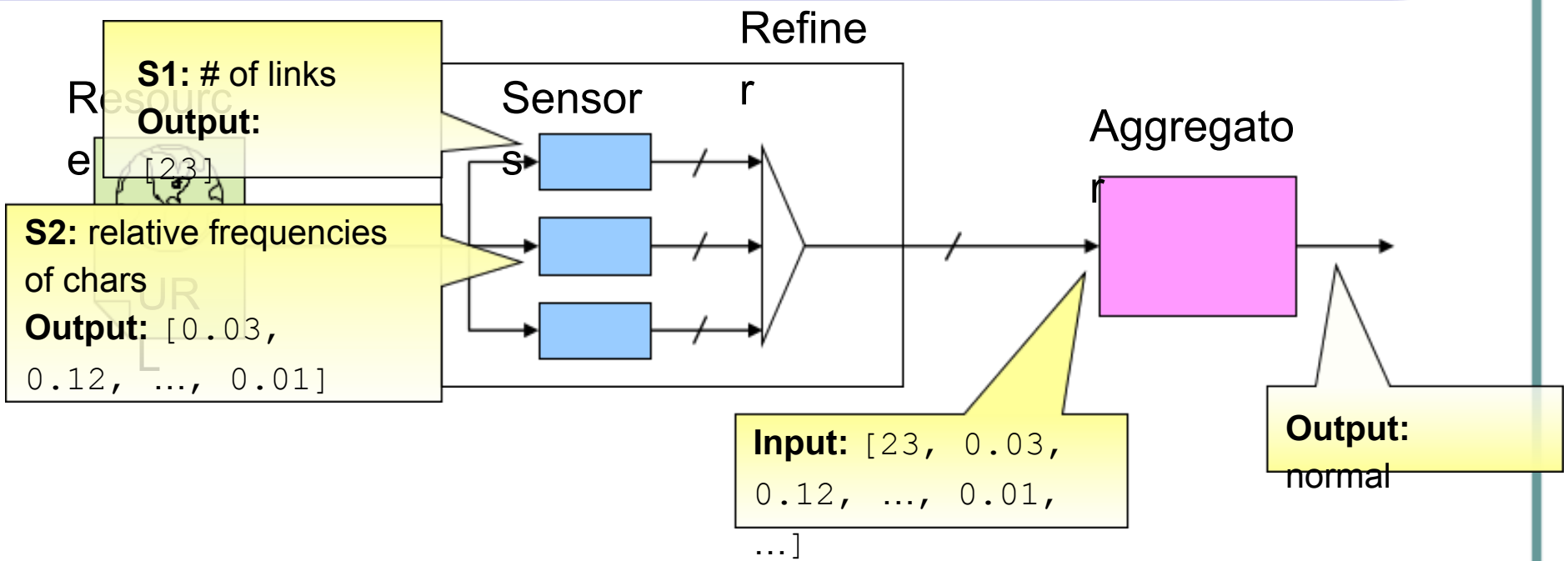
Scenario

- We base on an **Anomaly-based defacement detection** system that we developed earlier

  (IEEE Internet Computing Nov-Dec 2006)

  - Monitors **many remote** web pages at regular intervals and raises an alert when a page deviates from its profile
  - **Modular**

Detector: how it works



- Sensor
  - Functional block
  - Quantifies page features
- Aggregator
  - Builds the profile
  - Detects deviations from profile

Sensors

- In our prototype
  - 43 sensors
  - Grouped in 5 “categories”
  - Producing a numerical vector with 1466 elements
- “Many” page features from “many” points of view

Effectiveness indexes

- False positive rate (**FPR**)
- False negative rate (**FNR**)

GP in our tool (I)

- **Goal**: finding an optimal detection formula:
  - $f(v) = f(\{v_1, \ldots, v_{1466}\})$
  - $f(v) < 0 \Rightarrow$ negative reading

    $f(v) \geq 0 \Rightarrow$ positive (anomalous) reading
- With low FPR and low FNR

GP in our tool (II)

- **Terminals**:
  - subsets of elements of vector v output by the refiner, obtained with **feature selection**
  - constants {0, 0.1, 1}
- **Functions**: subsets of {+, -, *, /, unary-, min, max, ≤, ≥}
- **Fitness function**: f = FPR + FNR on the learning set
- **Stop criterion**: f=0 or 100 iterations done

Feature selection (I)

- We aim at selecting best vector elements
- **Key idea**: basing on absolute correlation
  - X_i: random variable of i-th element v_i
  - Y: random variable of the desired output (0 or 1)
  - c_i: absolute correlation of X_i with Y
  - $c_{i,j}$: absolute correlation of X_i with X_j

Feature selection (II)

- Iterate on:
  1. Select the element i with the highest c_i
  2. "Correct" the other elements: $c_{i,j} = c_{i,j} - c_i$

- In other words:
  1. Select the elements with the highest correlation with desired output but…
  2. … discard "duplicate" elements

Experiments: aggregators

- We compare the **GP** aggregator…
  - With 5 different **function sets**
  - With 5 different sizes for **feature selection**
- …to an **anomaly-based** aggregator that we developed earlier
  - Bases on domain knowledge
  - “Does not know” positive readings
- 25+1 aggregators

Experiments: data set

- Negative readings
  - **15 web pages** observed for…
  - …**1 month** and downloaded…
  - …every **6 hours**
  - totaling 125 reading for each page
- Positive readings
  - **75 readings** extracted from a publicly available attacks (defacements) archive

Experiments: methodology

- For each page, for each aggregator

1. We build a $S_{learning}$ with 50 negatives and 20 defacements
2. We build a $S_{testing}$ with 75 negatives and 75 defacements
3. We train the aggregator on $S_{learning}$
4. We compute FPR and FNR on $S_{testing}$

Experiments: results

Aggregator	FPR	FNR	f	n_g	t_s	t_h
Anomaly	1.42	0.09	-	-	-	-
GP-10-$\mathcal{F}_1$	0.00	0.71	0.0	1.0	17.0	2.7
GP-10-$\mathcal{F}_2$	0.09	0.98	0.0	1.0	23.3	3.7
GP-10-$\mathcal{F}_3$	0.09	0.62	0.0	1.1	20.4	3.5
GP-10-$\mathcal{F}_4$	4.53	0.44	0.0	1.0	20.7	3.7
GP-10-$\mathcal{F}_5$	0.09	0.89	0.0	1.0	27.9	3.9
GP-20-$\mathcal{F}_1$	0.09	1.16	0.0	1.0	18.2	2.6
GP-20-$\mathcal{F}_2$	0.18	1.33	0.0	1.0	12.8	2.4
GP-20-$\mathcal{F}_3$	0.36	0.80	0.0	1.0	20.1	3.1
GP-20-$\mathcal{F}_4$	0.09	0.89	0.0	1.0	36.5	4.2
GP-20-$\mathcal{F}_5$	0.00	0.89	0.0	1.0	39.5	3.9
GP-50-$\mathcal{F}_1$	0.00	1.24	0.0	1.0	5.1	1.6
GP-50-$\mathcal{F}_2$	0.09	0.98	0.0	1.0	20.4	2.9
GP-50-$\mathcal{F}_3$	0.36	0.98	0.0	1.0	19.3	2.9
GP-50-$\mathcal{F}_4$	0.18	0.89	0.0	1.0	15.4	3.1
GP-50-$\mathcal{F}_5$	**0.18**	**0.27**	**0.0**	**1.0**	**29.4**	**3.0**
GP-100-$\mathcal{F}_1$	0.09	1.16	0.0	1.0	15.5	2.1
GP-100-$\mathcal{F}_2$	0.09	1.33	0.0	1.1	11.4	2.2
GP-100-$\mathcal{F}_3$	0.00	1.87	0.0	1.3	14.1	3.1
GP-100-$\mathcal{F}_4$	0.09	0.27	0.0	1.1	18.7	3.1
GP-100-$\mathcal{F}_5$	0.09	1.33	0.0	1.2	15.4	2.6
GP-1466-$\mathcal{F}_1$	0.00	0.80	0.0	1.0	8.9	1.9
GP-1466-$\mathcal{F}_2$	0.18	0.44	0.0	1.0	6.1	2.3
GP-1466-$\mathcal{F}_3$	0.18	0.98	0.0	1.2	5.3	1.5
GP-1466-$\mathcal{F}_4$	0.18	1.87	0.0	1.2	11.2	1.8
GP-1466-$\mathcal{F}_5$	3.73	0.18	0.0	1.4	9.9	2.1

- GP performs better, but…
- … it is **never really engaged**
  - Only 1 generation
  - Trivial parse trees: $f(v) = 3v_{1233} - 15$
- Large attacker space!

Experiments 2: improvement

- **Key idea**: using both defacements and genuine readings of other pages as positives
- $S_{learning}$: 50 negatives + 20 defacements + 14 other pages readings
- $S_{testing}$: 75 negatives + 75 defacements + 70 other pages readings
- More demanding test

Experiments 2: results

Aggregator	FPR	FNR	f	n_g	t_s	t_h
Anomaly	1.42	2.39	-	-	-	-
GP-10-$\mathcal{F}_1$	9.87	2.48	0.4	35.5	83.8	7.2
GP-10-$\mathcal{F}_2$	9.24	1.84	0.0	14.3	69.1	7.5
GP-10-$\mathcal{F}_3$	9.24	1.29	0.1	35.7	66.4	8.0
GP-10-$\mathcal{F}_4$	11.38	1.98	0.1	24.1	93.1	7.9
GP-10-$\mathcal{F}_5$	7.73	1.52	0.1	30.5	66.1	6.9
GP-20-$\mathcal{F}_1$	23.38	2.30	0.1	16.9	54.2	5.3
GP-20-$\mathcal{F}_2$	16.44	1.61	0.0	10.8	68.3	6.2
GP-20-$\mathcal{F}_3$	13.51	1.33	0.0	16.4	52.1	6.2
GP-20-$\mathcal{F}_4$	17.87	1.38	0.0	14.6	56.3	6.3
GP-20-$\mathcal{F}_5$	17.87	0.55	0.0	19.5	70.4	6.5
GP-50-$\mathcal{F}_1$	14.76	1.56	0.1	23.7	43.1	4.4
GP-50-$\mathcal{F}_2$	13.16	1.61	0.0	4.7	45.0	5.4
GP-50-$\mathcal{F}_3$	18.58	0.83	0.0	11.7	32.4	5.4
GP-50-$\mathcal{F}_4$	7.56	1.52	0.0	12.5	41.1	6.0
GP-50-$\mathcal{F}_5$	11.47	1.66	0.0	16.1	71.2	6.8
GP-100-$\mathcal{F}_1$	0.62	2.30	0.0	29.4	51.5	4.5
GP-100-$\mathcal{F}_2$	5.51	1.38	0.0	10.5	25.6	4.1
GP-100-$\mathcal{F}_3$	6.93	0.55	0.0	16.4	33.9	4.8
GP-100-$\mathcal{F}_4$	5.87	1.61	0.0	10.2	40.3	5.1
GP-100-$\mathcal{F}_5$	12.09	1.52	0.0	17.7	31.9	5.2
GP-1466-$\mathcal{F}_1$	0.18	1.38	0.0	21.0	37.4	3.8
GP-1466-$\mathcal{F}_2$	0.44	1.10	0.0	18.5	24.8	4.1
GP-1466-$\mathcal{F}_3$	**0.71**	**0.64**	**0.0**	**15.1**	**30.1**	**4.7**
GP-1466-$\mathcal{F}_4$	5.69	1.06	0.0	19.7	27.9	4.7
GP-1466-$\mathcal{F}_5$	0.98	1.24	0.0	16.5	69.9	5.5

- GP still performs better…
- …and really works:
  - Many generations
  - Non trivial parse trees
- Feature selection is not necessary

Experiments: computation times

- Tuning procedure:
  - 100 s for GP aggregator (of which 5 sec in feature selection)
  - 10 ms for anomaly-based aggregator
- Single reading evaluation
  - 500 μs for GP
  - 100 μs for anomaly-based aggregator

- Questions?
- Thanks!