ArtForm

Dynamic analysis of JavaScript validation in web forms

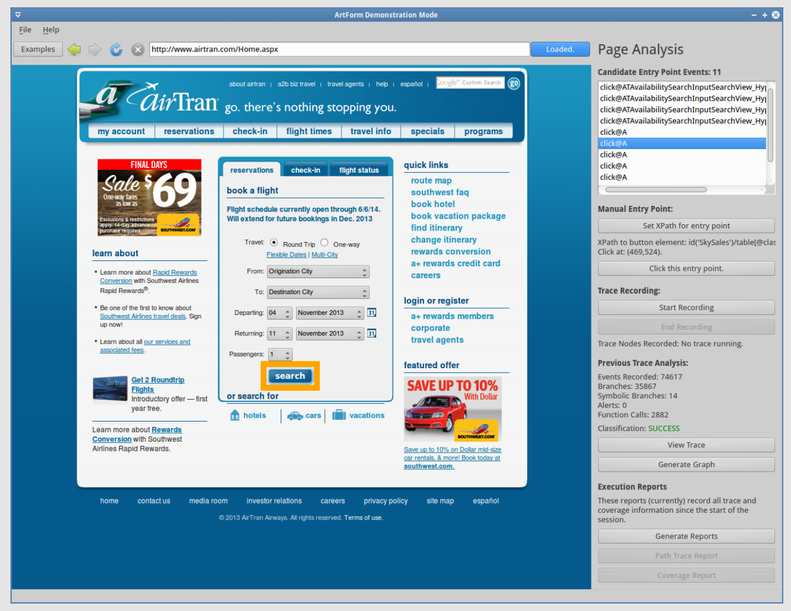
UNIVERSITY OF OXFORD

Microsoft Research

Form Understanding

When crawling the web, data is often hidden behind forms with validation constraints. We aim to analyse JavaScript code attached to forms and infer their integrity constraints, retrieve or model the hidden data, and improve the efficiency of searching or data-extraction tools.

- ▷ Dynamic features of JavaScript and real-world code make static analysis difficult.
- ▷ We use concolic analysis and perform a symbolic execution driven by concrete runs.
- ▷ ArtForm is built on Artemis[1], a tool for automated testing of JavaScript applications.



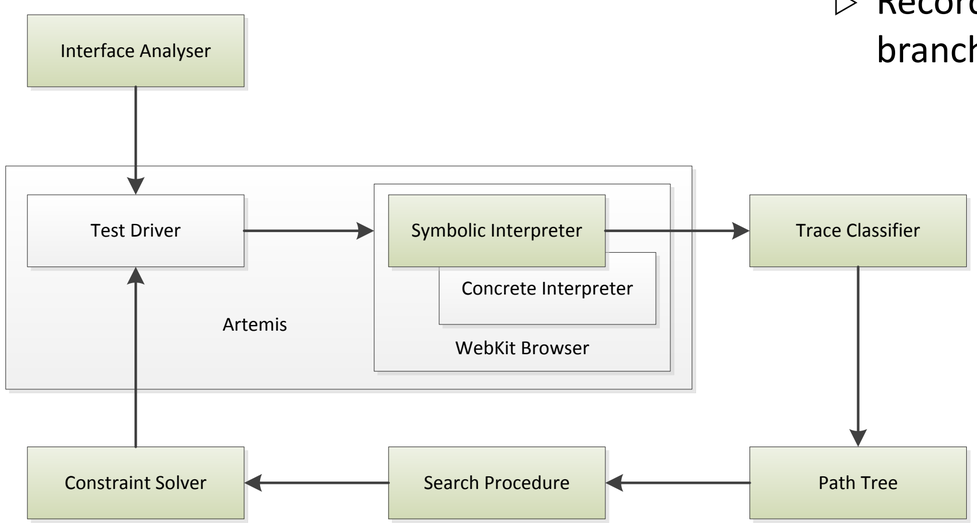
ArtForm Architecture

Interface Analyser

- ▷ Chooses the entry-point.
- ▷ Could be buttons, links, images or inputs and forms may cover the entire page.
- ▷ ArtForm uses DIADEM[2] for page analysis.

Test Driver

- ▷ Loads the test page with no saved state.
- ▷ Injects the given inputs into the form.
- ▷ Simulates real click on submit button.



Instrumented WebKit Browser

- ▷ Controls all code execution and events.
- ▷ Allows access to the DOM implementation and page information.

Symbolic Interpreter

- ▷ Tracks both concrete and symbolic values.
- ▷ Form inputs are initially symbolic variables and symbolic information is propagated as they are used.
- ▷ Records the path taken and the symbolic branches observed.

Trace Classifier

- ▷ Decides whether a trace was a successful submission or not.
- ▷ Based on alerts, changes to the DOM and page loads.

Path Tree

- ▷ Stores information about the previous runs.

Constraint Solver

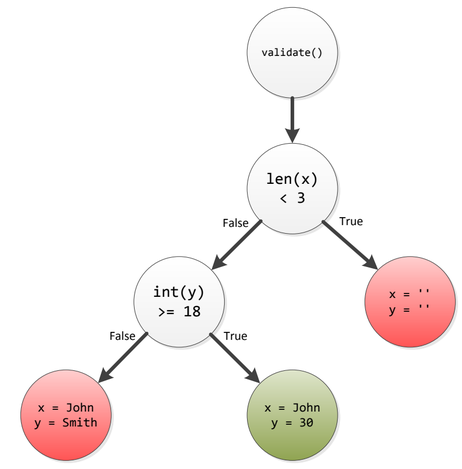
- ▷ Solves the path constraint to generate the next input values to test.
- ▷ Translates our internal constraints to input for a third-party solver.
- ▷ ArtForm uses CVC4[3].

Search Procedure

- ▷ Chooses the next path to explore.
- ▷ We are Investigating good search strategies to find interesting parts of the tree more quickly.

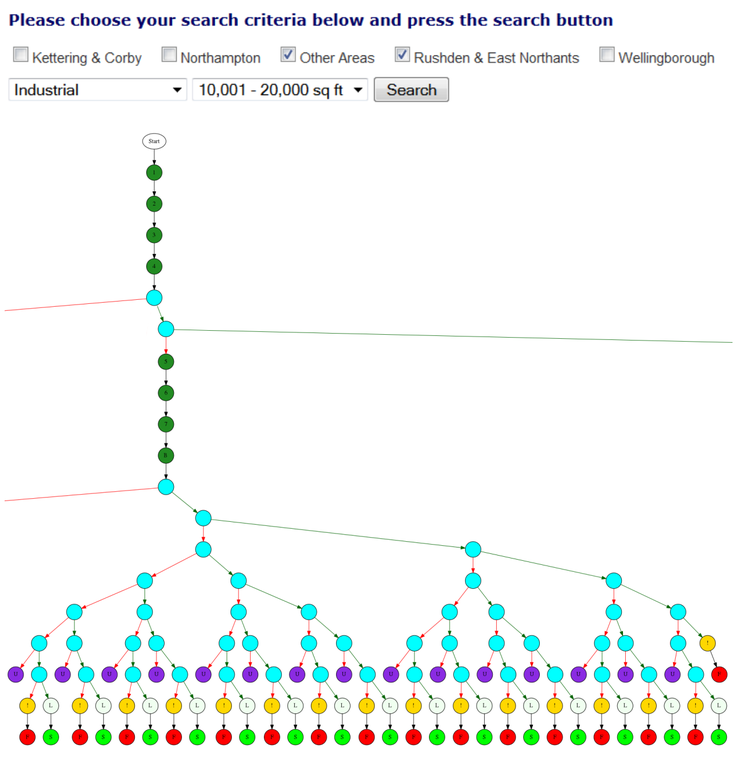
Concolic Analysis Example

```
function validate() {
    x = document.getElementById("name").value;
    y = document.getElementById("age").value;
    if (x.length < 3) {
        alert("Error!"); return false;
    } else {
        if (parseInt(y, 10) >= 18) {
            return true;
        } else {
            alert("Error!"); return false;
        }
    }
}
```



Run	x	y	Conditions	Valid	Next Goal
1	''	''	`len(x) < 3`	No	`len(x) >= 3`
2	'John'	'Smith'	`len(x) >= 3` `int(y) < 18`	No	`len(x) >= 3` `int(y) >= 18`
3	'John'	'30'	`len(x) >= 3` `int(y) >= 18`	Yes	Finished

Web Form Example



URL: www.underwoods.co.uk/search.cfm

Branches: 629	Success: 248	Failure: 164
Constraints: 692	Traces: 412	Unsatisfiable: 280

Issues

- ▷ Event handlers
  - ▷ Each field has its own validation function and may depend on other fields.
  - ▷ Triggering them in-order is a useful guess, but may miss some information.
  - ▷ Checking dependence between the handlers is difficult.
  - ▷ Testing all orderings is not feasible.
- ▷ Values may be tested before injection.
- ▷ Forms may be updated dynamically.
- ▷ Select boxes and radio buttons have implied constraints found in the DOM.
- ▷ JavaScript's coercion semantics and NaN are difficult to model and solve.
- ▷ Some constraint types are not supported by the solver or our translation.
- ▷ JavaScript minification and obfuscation.
- ▷ Long loops and repeated code.

Future Work

- ▷ Continue testing on real-world sites to identify common patterns which we do not handle well.
- ▷ Track event handler dependencies and use partial order methods to choose appropriate orderings to test.
- ▷ Use heuristics or static analysis to target the search and speed up exploration of the most interesting parts of the trees.
- ▷ Work on converting the trees to useful descriptions of the constraints.
- ▷ Infer higher-level constraints such as set containment which are not shown directly in our traces.
- ▷ New features for our translator and the solver, for example more complex regular expression tests or string inequalities.

Ben Spencer, supervised by Michael Benedikt and Franck van Breugel
In collaboration with Casper Svenning Jensen and Anders Møller

ArtForm is available at github.com/cs-au-dk/Artemis

[1] brics.dk/artemis/
[2] diadem.cs.ox.ac.uk
[3] cvc4.cs.nyu.edu/web/