System Level Metamorphic Testing & GUI Automation Hands-on

CS453 Automated Software Testing

Shin Yoo

Testing Search Engines

- Understandably complicated task
- A primary component is human raters, who rate URLs good or bad
- If a new change in search algorithm results in bad URLs moving up w.r.t. the same query string, the change becomes a problem
- Can we do better?

Metamorphic Testing for Search Engines

- Not the final answer to testing, as MT generally is, but a helping hand nonetheless.
- Metamorphic Testing for Software Quality Assessment: A Study of Search Engines, Zhi Quan Zhou, Shaowen Xiang, and Tsong Yueh Chen, TSE 42(3):264-284, 2016

MRs Explained

- MPSite: if original query Q finds page P in domain D, the follow-up query “Q site:D” should still find page P
- MPTitle: if P and Q are known synonyms, queries P and Q should produce the same set of pages
- MPReverseJD: if P is a set of query terms, and Q is the reverse of P, they should return the same set of pages

Let’s add our own

- MPSpecific: if query P returns X pages, the follow up query P AND Q should return Y pages where Y <= X

How do we automate this?

- This is the top level system testing. We would like to go through the GUI to emulate the user experience.
- We will briefly cover the evolution of GUI automation tools.

Capture and Replay

- Record system events by injecting event handler hooks
- Replay later
- Pro
  - Intuitive, simple, automates the most tedious part
- Con
  - Can be fragile, as the capability to perform replay depends on how the tool records the events

UI Scripting

- Allows identification of UI elements using internal information (e.g., XPath, DOM)
- Probably the current mainstream (Selenium)
- Pro
  - Precise automation
- Con
  - You have to know the code to write automation
  - Can still break (although rarer than capture and replay)

Visual Automation

- Lauded as the next generation automation method
- Use computer vision to recognise UI elements: no need for the knowledge of internal structures
- Pro
  - Intuitive, can be written by non-developers
- Con
  - Possibly fragile again (graphical elements may change), but can be tied to resources systematically
  - Computationally expensive

Tools

- Selenium https://www.selenium.dev (Our focus today)
  - Can drive popular browsers automatically
  - Has driver wrappers for many languages
- Apparition https://github.com/twalpole/apparition
  - A headless driver that works in conjunction with Capybara (UI automation language)
- SikuliX https://sikulix.github.io
  - Visual automation tool (demo)

Goal

- To hack a test script for our MR for Google Search
  - Implement an end-to-end metamorphic test case using the MPSpecific metamorphic relationship
  - First, make a Google search query P, and store the number of pages returned, X
  - Second, make a search query P and Q, and store the number of pages returned, Y
  - Check X >= Y

Selenium: Installation

- The easiest way for today's hack would be to use Python wrapper and a driver for your main web-browser
  - You can do pip install selenium
  - Plus you need an executable that will drive your choice of web browser. See https://selenium-python.readthedocs.io/installation.html#drivers for download links; you need to put the executable on your PATH
  - Windows users: https://selenium-python.readthedocs.io/installation.html#detailed-instructions-for-windows-users

Selenium Starting Point

```
from selenium import webdriver
from selenium.webdriver.common.keys import Keys

driver = webdriver.Chrome()
driver.get("http://www.python.org")
assert "Python" in driver.title
elem = driver.find_element_by_name("q")
elem.clear()
elem.send_keys("pycon")
elem.send_keys(Keys.RETURN)
assert "No results found." not in driver.page_source
driver.close()
```

… and then see: https://selenium-python.readthedocs.io/index.html