Verification Tools in Practice

Testing, Quality Assurance, and Maintenance
Winter 2017

Prof. Arie Gurfinkel

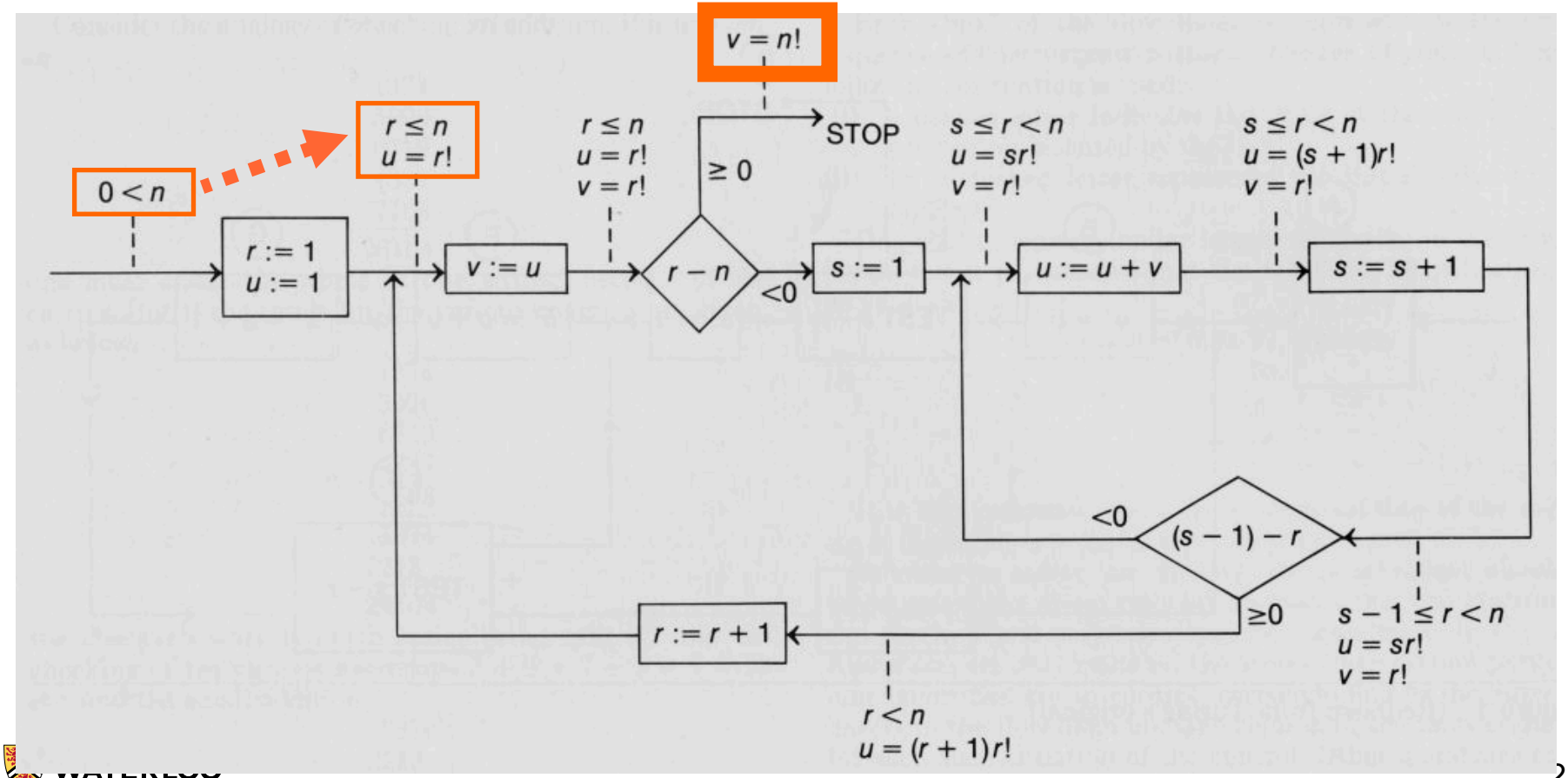
UNIVERSITY OF
WATERLOO

Turing, 1949

Alan M. Turing. “Checking a large routine”, 1949

How can one check a routine in the sense of making sure that it is right?

programmer should make a number of definite assertions which can be checked individually, and from which the correctness of the whole programme easily follows.



2

Verification Competition

http://etaps2016.verifythis.org/

Microsoft Visual Studio Products

Code Contracts

- https://marketplace.visualstudio.com/items?itemName=RiSEResearchinSoftwareEngineering.CodeContractsforNET
- https://github.com/Microsoft/CodeContracts

- statically and dynamically checked method pre- and post-conditions

IntelliTest

- https://www.visualstudio.com/en-us/docs/test/developer-testing/intellitest-manual/introduction

- automated test generation by dynamic symbolic execution

WHY3

http://why3.lri.fr/

VeriFast

https://people.cs.kuleuven.be/~bart.jacobs/verifast/

Viper

http://www.pm.inf.ethz.ch/research/viper.html

Open JML

http://www.openjml.org/

The KeY Project

https://www.key-project.org/

Proving that Android's, Java's and Python's sorting algorithm is broken (and showing how to fix it)

February 24, 2015 Envisage Written by Stijn de Gouw. $s

Tim Peters developed the Timsort hybrid sorting algorithm in 2002. It is a clever combination of ideas from merge sort and insertion sort, and designed to perform well on real world data. TimSort was first developed for Python, but later ported to Java (where it appears as java.util.Collections.sort and java.util.Arrays.sort) by Joshua Bloch (the designer of Java Collections who also pointed out that most binary search algorithms were broken). TimSort is today used as the default sorting algorithm for Android SDK, Sun's JDK and OpenJDK. Given the popularity of these platforms this means that the number of computers, cloud services and mobile phones that use TimSort for sorting is well into the billions.

http://envisage-project.eu/proving-android-java-and-python-sorting-algorithm-is-broken-and-how-to-fix-it/

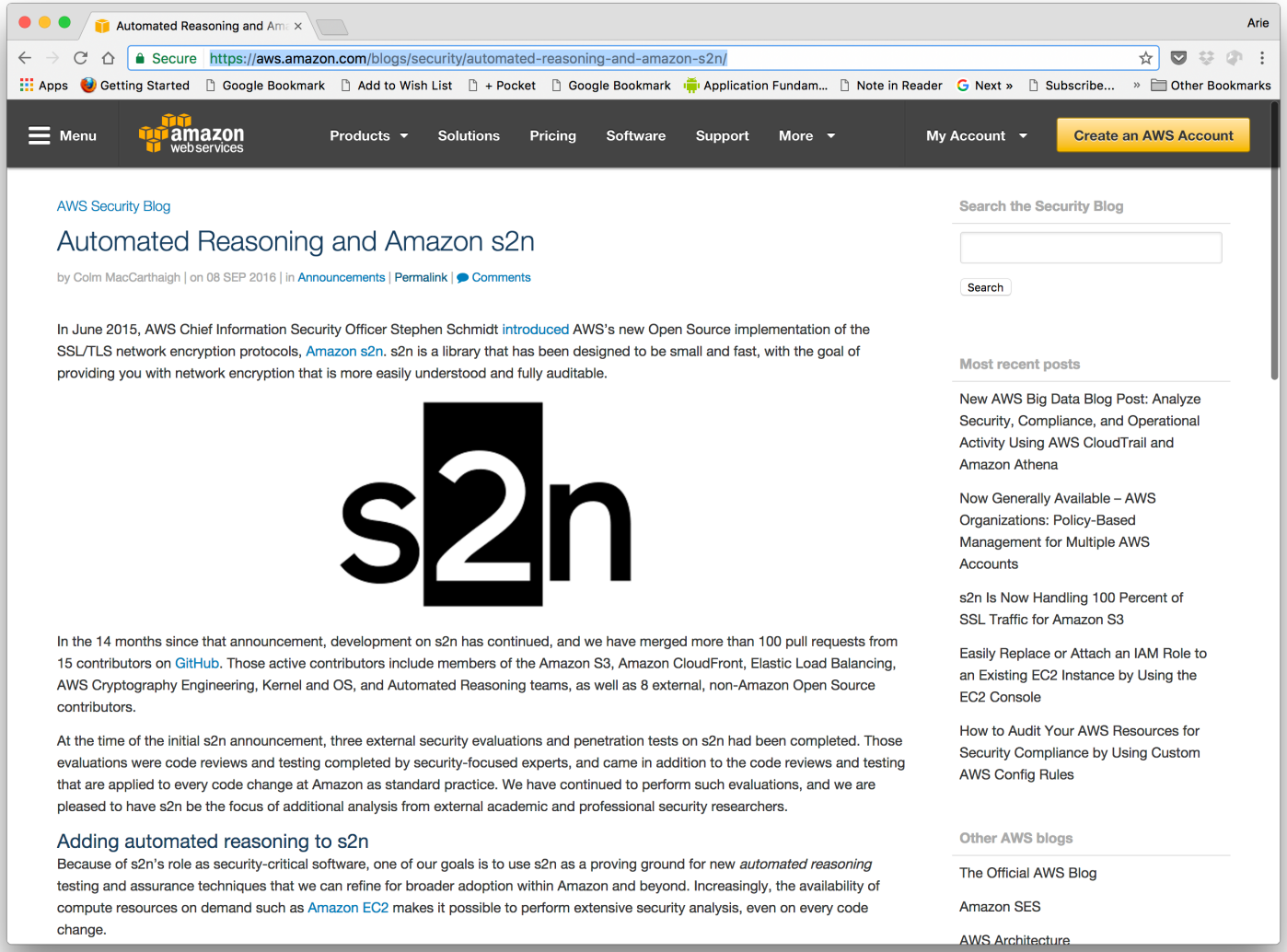
Frama-C

https://frama-c.com/

SPARKPro

http://www.adacore.com/sparkpro/

Amazon S2N



https://aws.amazon.com/blogs/security/automated-reasoning-and-amazon-s2n/

IronClad and InronFleet

https://github.com/Microsoft/Ironclad

Is Verification Enough

Can verified software fail?

Do we need both testing and verification?