

Microsoft Research @ CAV'21

- Nikolaj Bjorner
- Leonardo de Moura
- Shuvendu Lahiri
- Akash Lal
- Andrey Rybalchenko
- Nikhil Swamy

Overview

- Projects
 - Concurrency Unit Testing with Coyote // Akash
 - F* // Nikhil
 - Network Verification // Andrey & Nikolaj
 - Merge Conflict Resolution // Shuvendu
- Career opportunities/Outreach // Jinoos
- Q&A

Coyote – Concurrency Unit Testing

Motivation

- Concurrency is a fundamental to building scalable systems
- However, traditional testing techniques are often ineffective in finding or reproducing concurrency bugs
- Vision: Every developer should be able to write **concurrency unit tests** as simply as writing regular unit tests

What is Coyote?

A .NET library and tool for writing **concurrent unit tests** in C#

- APIs for writing unit tests, specifications and capturing sources of nondeterminism
- Takes control of the task execution during testing
- Scheduler that enumerates task interleavings and deterministically replays bugs

Released as open-source on GitHub

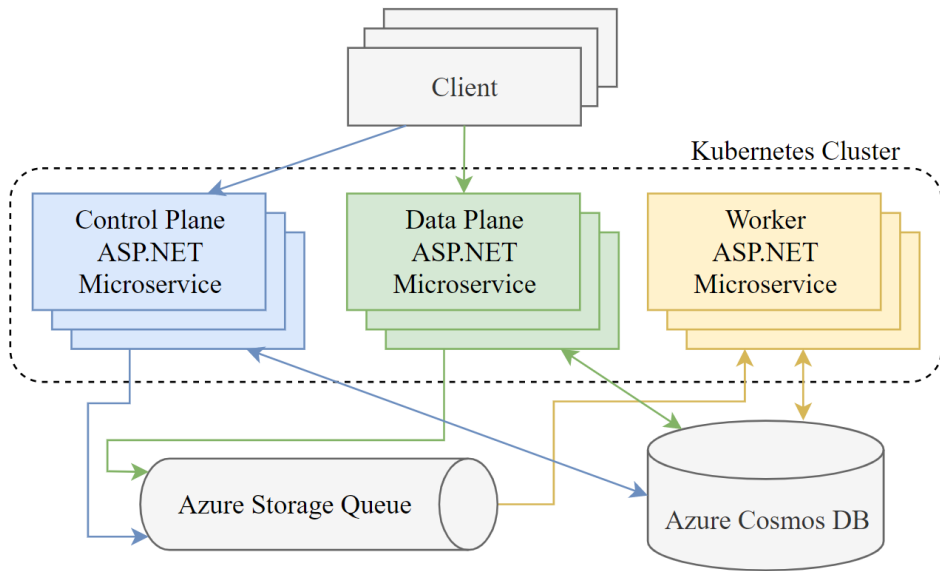
- <https://github.com/microsoft/coyote>
- <https://www.nuget.org/packages/Microsoft.Coyote/>

↓ 711,135 total downloads

📦 146 downloads of current version

📈 1,130 downloads per day (avg)

Testing Azure Services



Used routinely to test 15+ distributed Azure services

C# Concurrent Unit Test

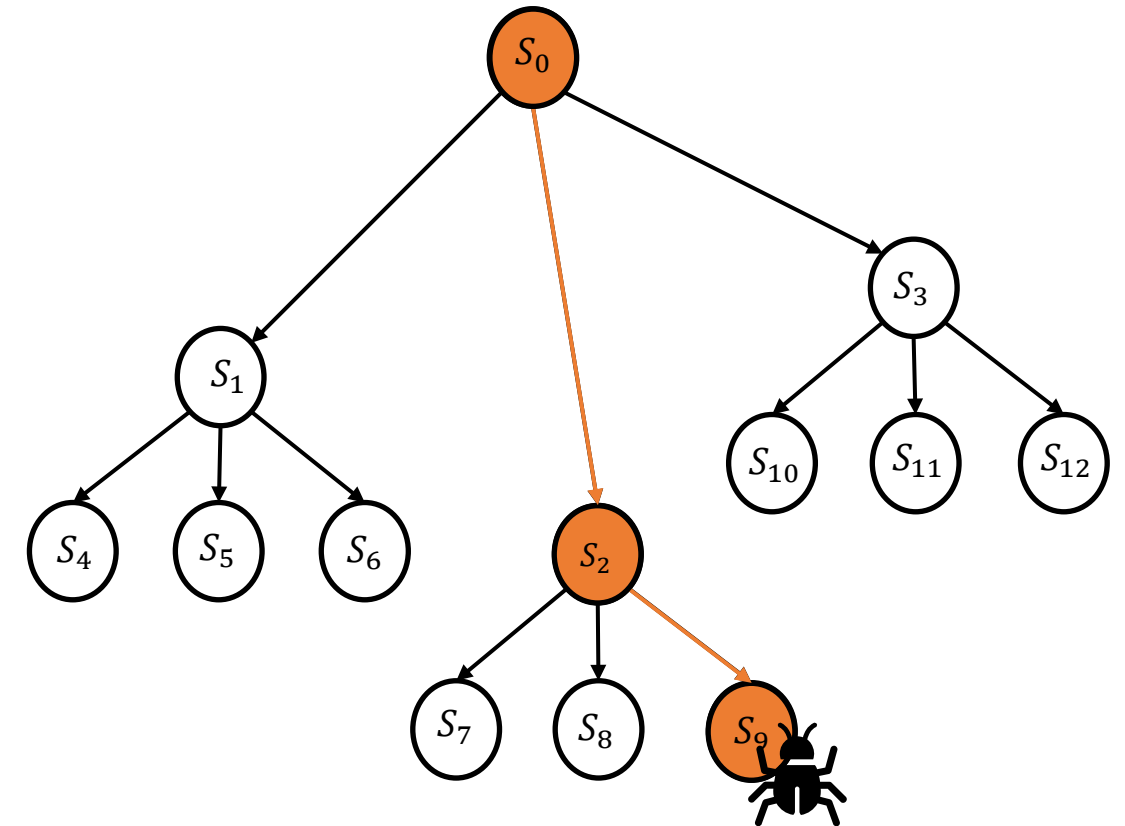
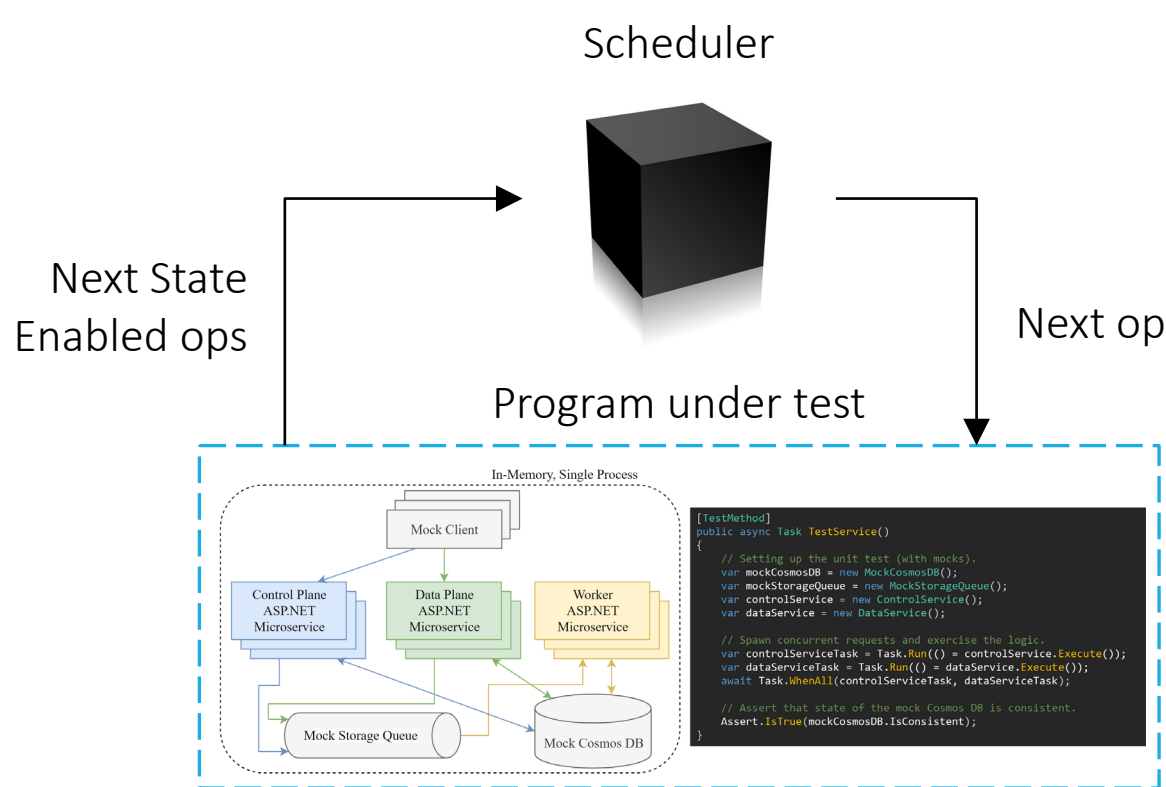
```

[TestMethod]
public async Task TestService()
{
    // Setting up the unit test (with mocks).
    var mockCosmosDB = new MockCosmosDB();
    var mockStorageQueue = new MockStorageQueue();
    var controlService = new ControlService(mockCosmosDB, mockStorageQueue);
    var dataService = new DataService(mockCosmosDB, mockStorageQueue);

    // Spawn concurrent requests and exercise the logic.
    var controlServiceTask = Task.Run(() => controlService.Execute());
    var dataServiceTask = Task.Run(() => dataService.Execute());
    await Task.WhenAll(controlServiceTask, dataServiceTask);

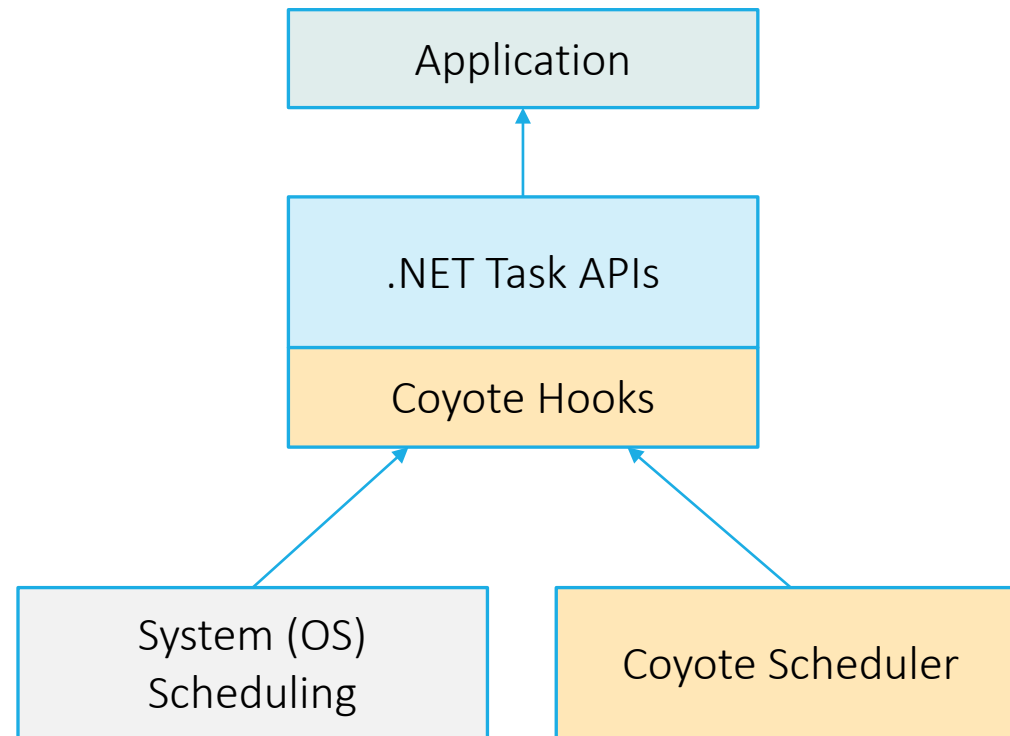
    // Assert that state of the mock Cosmos DB is consistent.
    Assert.IsTrue(mockCosmosDB.IsConsistent);
}
  
```

Algorithms for coverage

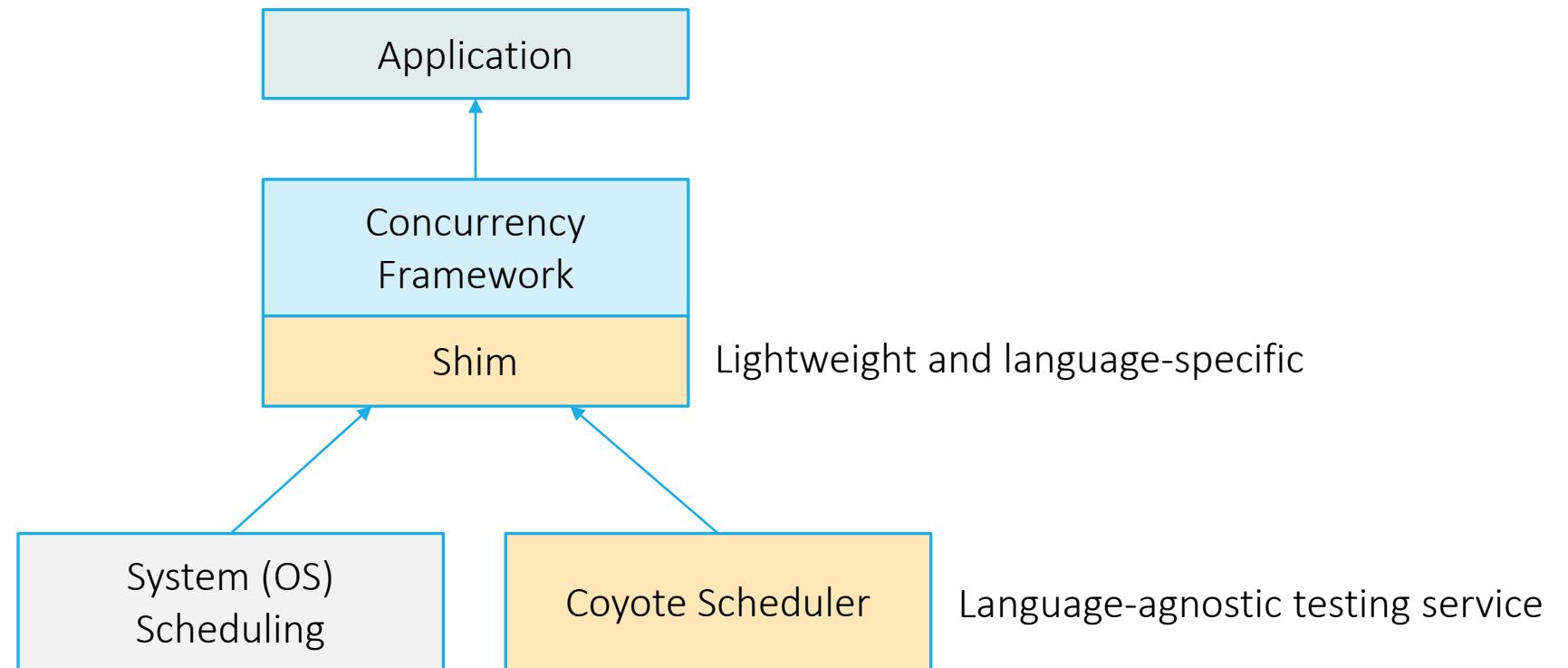


Learning Based Controlled Concurrency Testing. OOPLSA 2020. *Distinguished Artifact Award*.

Concurrency unit testing for all



Concurrency unit testing for all



Nekara: Generalized Concurrency Testing. ASE 2021.

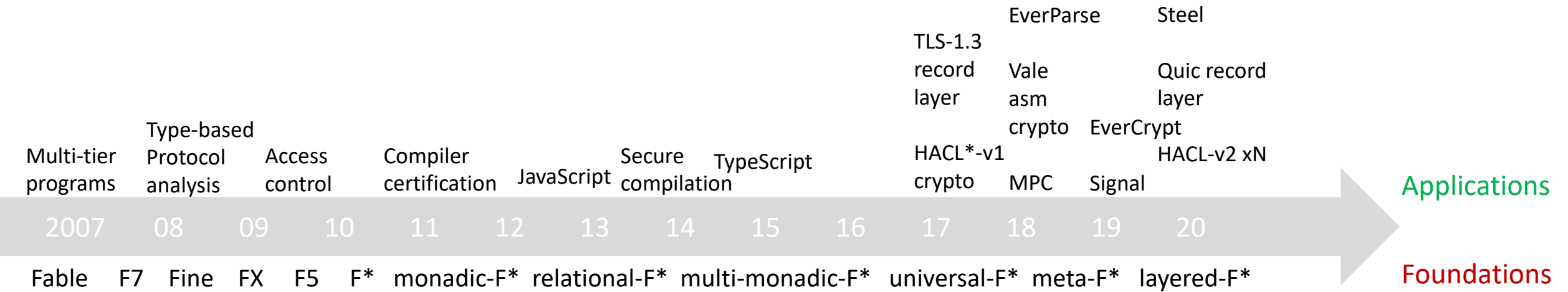
Coyote

<https://github.com/microsoft/coyote>

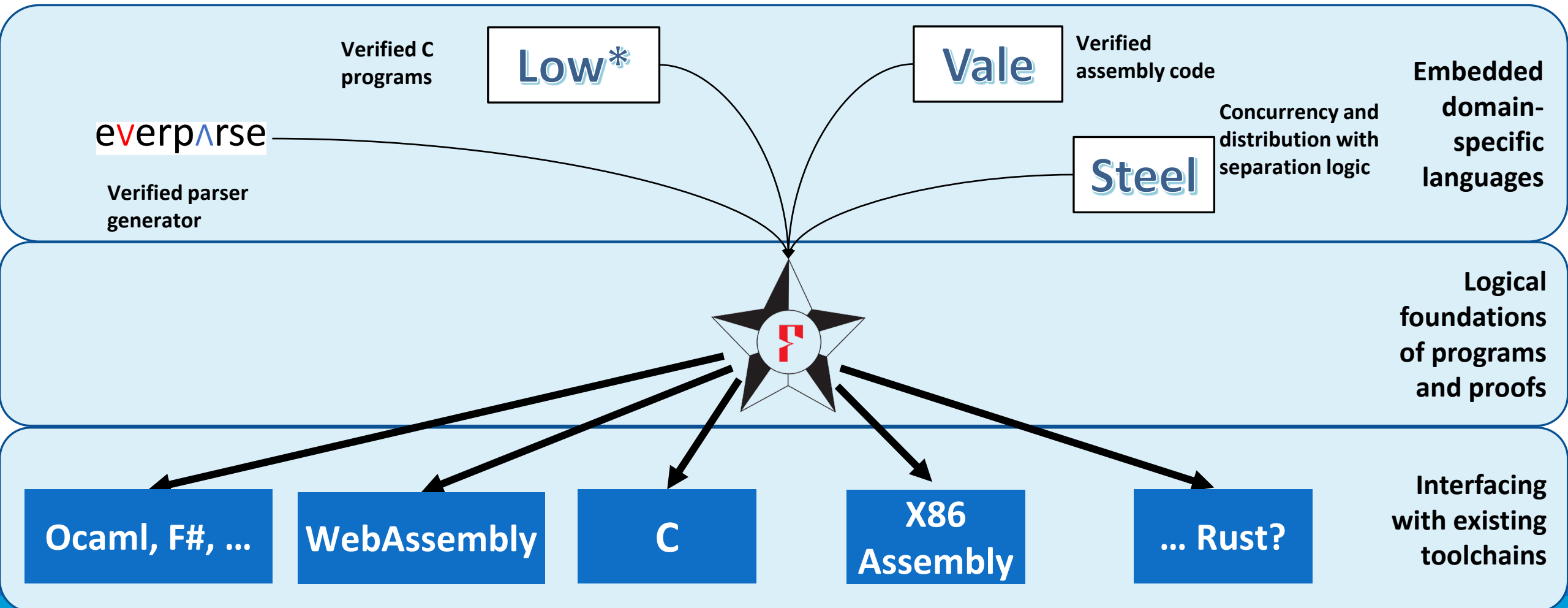
Builds on years of research, to offer a competitive edge when developing distributed services – improved productivity, less bugs in production – adopted by >15 services in Azure

Video summary of Coyote: [Developer Tech Minutes: Project Coyote - YouTube](#)

F*: A Programming Language and Proof Assistant



F*: A Programming Language and Proof Assistant



Program proofs in F* for billions of unsuspecting users

Automated parsing
untrusted data with proofs
in Hyper-V/VMSwitch



Verified
cryptography in the
Linux kernel

Verified Merkle trees
for Enterprise
blockchains



Quic transport,
MSQuic in Windows,
Verified crypto in Firefox,
mbedtls,
Signal in Wasm,
Wireguard, ...



High-value domains
Fintech, verified

Fintech companies
investing in and
contributing to F*

Core crypto in
ElectionGuard

Algorithm	Portable C (HACL*)	Intel ASM (Vale)	Agile API (EverCrypt)
AEAD			
AES-GCM		✓ (AES-NI + CLMUL)	✓
Chacha20-Poly1305	✓ (+ AVX,AVX2)		✓
ECDH			
Curve25519	✓	✓ (BMI2 + ADX)	
P-256	✓		
Signatures			
Ed25519	✓		
P-256	✓		
Hashes			
MD5	✓		✓
SHA1	✓		✓
SHA2-224,256	✓	✓ (SHAEXT)	✓
SHA2-384,512	✓		✓
SHA3	✓		
Blake2	✓ (+ AVX,AVX2)		
Key Derivation			
HKDF	✓	✓ (see notes below)	✓
Ciphers			
Chacha20	✓ (+ AVX,AVX2)		
AES-128,256		✓ (AES-NI + CLMUL)	
MACS			
HMAC	✓	✓ (see notes below)	✓
Poly1305	✓ (+ AVX,AVX2)	✓ (X64)	

EverCrypt

A verified industrial-grade cryptographic provider.

A replacement for: OpenSSL, Bcrypt, libsodium.

- A *collection* of algorithms (**exhaustive**)
- Easy-to-use API (**CPU auto-detection**)
- Several *implementations* (**multiplexing**)
- APIs grouped by *family* (**agility**)

Clients get **state-of-the art performance**.

- 130,000 lines of Low* and 24,000 lines of Vale (F* DSLs)
- 65,000 lines of C + 15,000 lines of ASM



Azure CCF

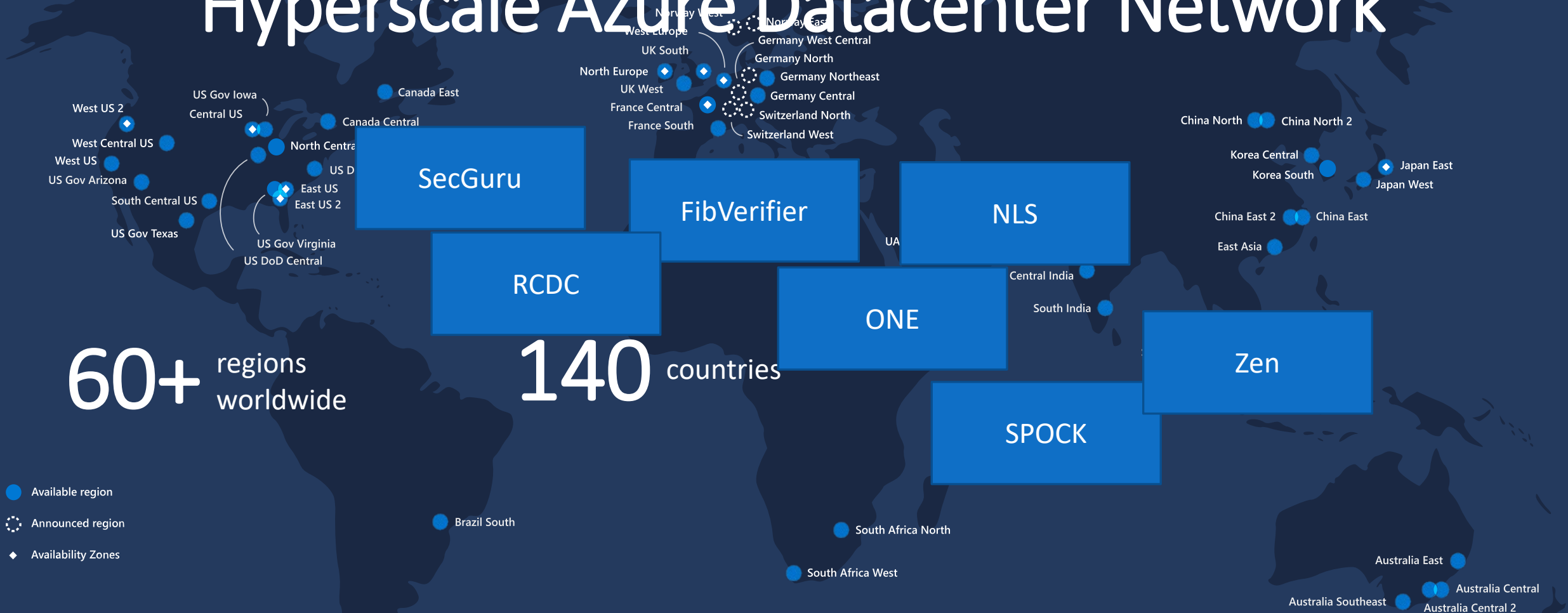
DICE



ElectionGuard



Hyperscale Azure Datacenter Network



60+ regions worldwide

140 countries

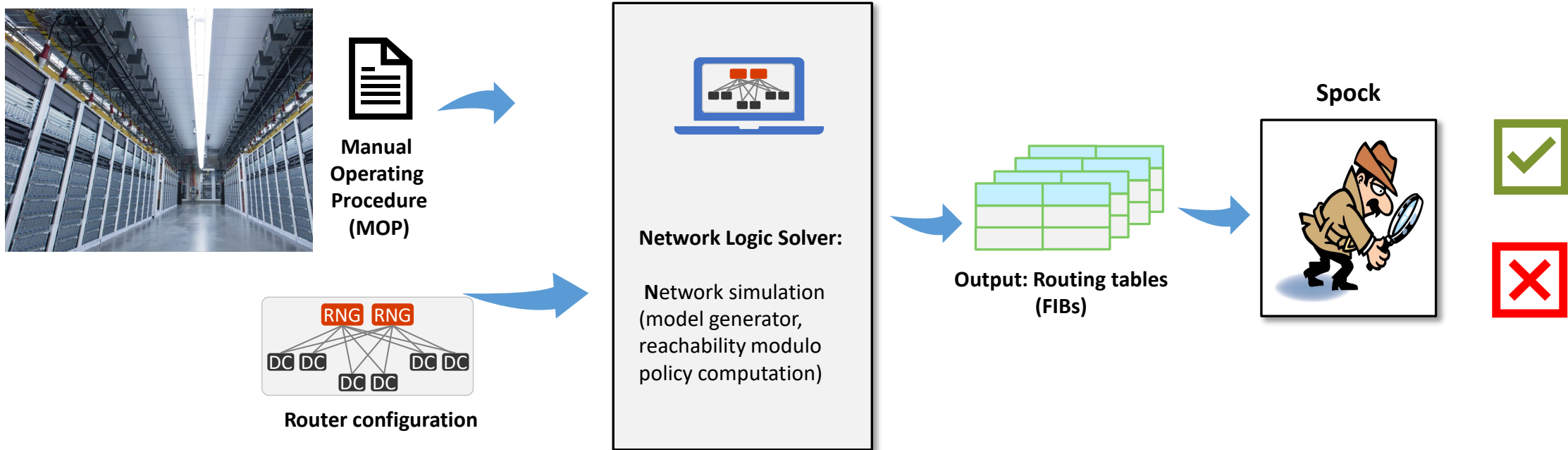
$O(100K)$ network devices

$O(1M)$ virtual networks

$O(1M)$ policies

$O(1K)$ maintenance changes/day

Network Change Verification System (NCVS) for Azure Datacenters





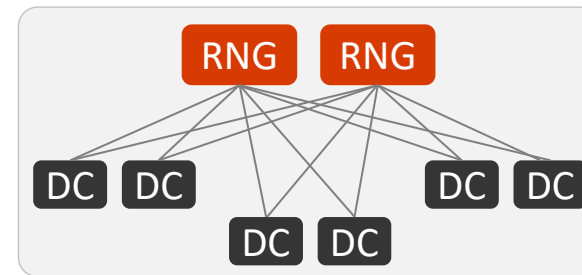
Network Logic Solver

Network Verification at Azure Scale

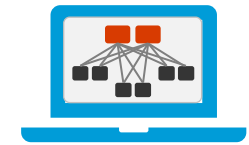
- Simulates the control plane (BGP, IS-IS, RSVP-TE, MPLS)
- Vendor specific protocol semantics for Cisco, Arista, Juniper, SONIC
- High performance (50,000 routers in 10min)
- High fidelity (via daily production equivalence check)

How

- “Explicit state” model generator
- Global model of distributed routing protocols
- Highly optimized data-structures

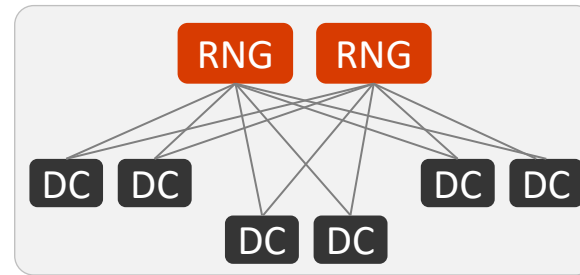


router configurations



Network Logic Solver

Network Verification @ Azure



router configurations



Network Logic Solver

Spock Architecture

```
[SpockTestMethod(scope: Datacenter)]
public void TorReachabilityTest(Datacenter dc) {
    foreach (var t0 in dc.T0s) {
        foreach (var prefix in t0.VlanAddresses) {
            var relevant = ContainedBy(prefix);
            var reachable = Reachable(dc.T0s, t0);
            var invariant = If(relevant, reachable);
            Assert.AllPackets(invariant);
            ...
        }
    }
}
```

Input: Specification as a test



Input: Azure region topology



Input: Routing tables (FIBs)



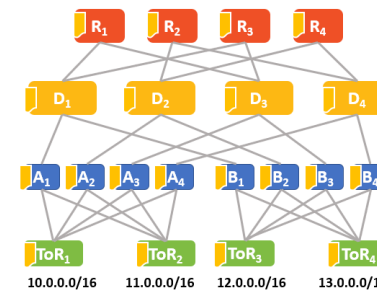
Spock = Zen + configuration adapters

Open source

```
Passed: BgpSessionTest
Failed: TorReachabilityTest
Datacenter: cpq01
Packet: 70.12.1.3
Path: cpq01-0108-0110,
cpq01-0108-12t1
...
```

Output: Test result

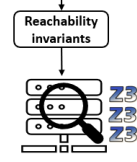
Live Monitoring of Forwarding Behavior



Global reachability as **local contracts**



Topology Database



- ✓ Each router has a fixed rule for a set of addresses
- ✓ Enough to verify rule is enforced on each router

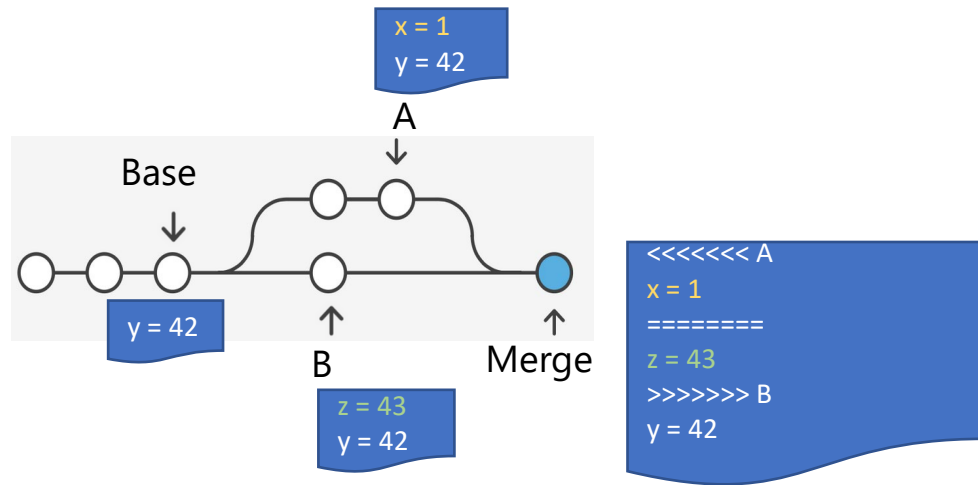


Error Reports

5 Billion Z3 queries per day

[Jayaraman et al, [Sigcomm 2019](#)]

Merge Conflicts: bane for code collaboration!



"Among those [challenges in Pull-Based Development], the use of git and **handling conflicts between branches are the most prominent ones**, especially **for seemingly less experienced developers**" [1]

- **Frequent:** 10-20% of all merges fail with conflicts^[2,3]
 - Scalable merge algorithms cannot accommodate program semantics
 - Includes spurious textual conflicts, and silent semantic conflicts that introduce bugs
- **Non-trivial resolution cost**
- **Effect:** Hurts **team productivity** and **reliability**
 - Stalls CI/CD pipelines
 - Introduces bugs and vulnerabilities
 - Worsens super-linearly with team size^[2]

[1] Work Practices and Challenges in Pull-Based Development: The Contributor's Perspective, Gousios et al., ICSE'16

[2] G. Giotto, et al., "On the Nature of Merge Conflicts: A Study of 2,731 Open Source Java Projects Hosted by GitHub," in IEEE Transactions on SW Engineering 2020

[3] Y. Brun, R. Holmes, M. D. Ernst, and D. Notkin, "Proactive detection of collaboration conflicts," ESEC/FSE, 2011

Project DeepMerge

Resolve syntactic and semantic merge conflicts at scale leveraging complementary powers of symbolic reasoning and deep learning

Formally verified merge

- *Program Integration* problem from 30 year back
 - **Semantic-conflict-freedom**: Any behavior change introduced by A or B is preserved, and no other new behavior is introduced
- Revisiting after ~30 years with advances in SMT/Verification
 - Sousa, Dillig, Lahiri: *Verified three-way program merge*. OOPSLA 2018
 - Good news for verification at scale
 - (Specification) Semantic conflict freedom can be automatically stated as an assertion over states of 4 programs (O, A, B, M)!
 - (Scale) Verification can scale with the size of the changes
 - (Invariants) Simple class of relational invariants suffice for most programs
- Challenges
 - How to synthesize M?

Integrating non-interfering versions of programs

Authors Susan Horwitz, Jan Prins, Thomas Reps

Publication date 1989/7

Journal ACM Transactions on Programming Languages and Systems (TOPLAS)

We say that \mathcal{M} is *semantically conflict-free*, if for all valuations σ such that:

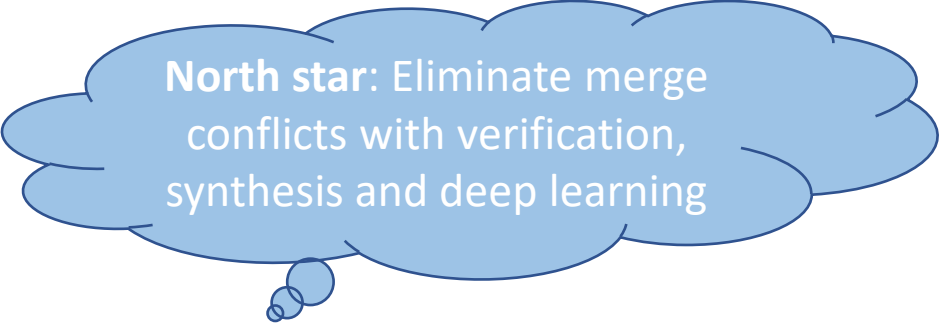
$$\sigma \vdash O \Downarrow \sigma_O \quad \sigma \vdash \mathcal{A} \Downarrow \sigma_{\mathcal{A}} \quad \sigma \vdash \mathcal{B} \Downarrow \sigma_{\mathcal{B}} \quad \sigma \vdash \mathcal{M} \Downarrow \sigma_{\mathcal{M}}$$

the following conditions hold for all i :⁴

- (1) If $\sigma_O[(out, i)] \neq \sigma_{\mathcal{A}}[(out, i)]$, then $\sigma_{\mathcal{M}}[(out, i)] = \sigma_{\mathcal{A}}[(out, i)]$
- (2) If $\sigma_O[(out, i)] \neq \sigma_{\mathcal{B}}[(out, i)]$, then $\sigma_{\mathcal{M}}[(out, i)] = \sigma_{\mathcal{B}}[(out, i)]$
- (3) Otherwise, $\sigma_O[(out, i)] = \sigma_{\mathcal{A}}[(out, i)] = \sigma_{\mathcal{B}}[(out, i)] = \sigma_{\mathcal{M}}[(out, i)]$

Synthesize candidate merges: some progress

- Program synthesis
 - Design DSLs to learn repeated patterns in large projects using PROSE
 - Pan, Le, Nagappan, Gulwani, **Lahiri**, Kaufman: *Can Program Synthesis be Used to Learn Merge Conflict Resolutions? An Empirical Analysis*. ICSE 2021
- Machine learning for code
 - Learn to perform merge from manual user resolutions on GitHub
 - Dinella, Mytkowicz, Svyatkovskiy, Bird, Naik, **Lahiri**: *DeepMerge: Learning to Merge Programs*. arXiv 2105.07569



North star: Eliminate merge conflicts with verification, synthesis and deep learning

Microsoft Research, Outreach - Programs & Opportunities

Global PhD Fellowship Program – [Academic Programs - Microsoft Research](#)

- Microsoft collaborates with the global research community through programs, events, learning opportunities, and joint research endeavors.
- We're currently reviewing this years' applicants. We will be opening the next call for submissions in May 2022. To apply, please go to the website directed above or email msfellow@microsoft.com.

Microsoft Research Blogs, Podcasts & Webinars

- Join us for a webinar, podcast, or get to know our researchers through our blogs all posted on to our Microsoft Research website - [Microsoft Research – Emerging Technology, Computer, and Software Research](#)
- You can subscribe to our [newsletter](#) to gain awareness into research projects, upcoming events and conferences, and any career opportunities.

3rd Party Conferences – Programming Languages

Microsoft Research is committed and focused on sponsoring all Programming Languages conferences through virtual booth platforms, accepted papers, posters, and talk sessions. You can find us to chat with our researchers and/or recruiters at any of the following top research conferences this next fiscal year, and many others in between:

- [PLDI](#) Conference
- [CAV](#) Conference
- [POPL](#) Conference
- [PPoPP](#) Conference
- [ICFP](#) Conference

Further inquiries, please email Jinoos Safavian at jisafa@microsoft.com.