Use of formal methods at AWS

Setting the expectations

- Basic concepts of *Formal Methods*
- We go one step further than the paper
- No TLA+ (sorry!), but PROMELA

Amazon Web Services

- Cloud computing service provider
- Largest market share in public cloud services (Almost 1/3rd!)
- An example architecture..

In Drump’s words

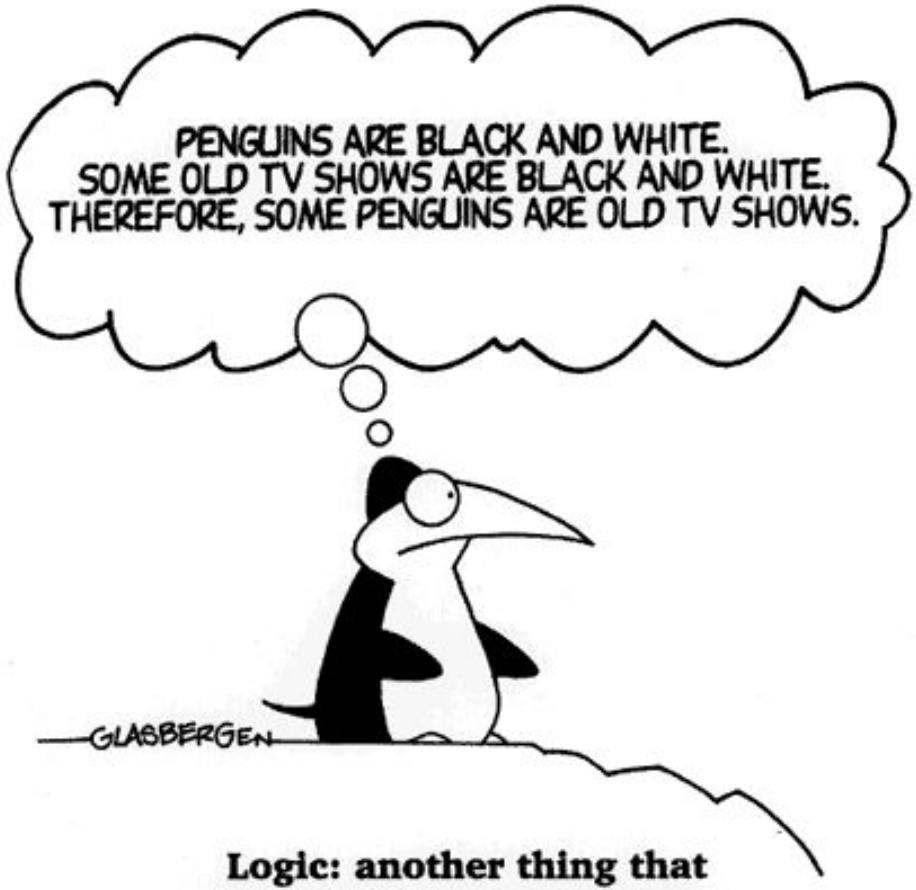
Lots of concurrency… Lots of distribution... Great stuff. Absolutely great.

What is ‘Formal Methods’?

*"Formal Methods" refers to **mathematically rigorous** techniques and tools for the specification, design and verification of software and hardware systems.*[1]

How rigorous? Machine verifiable.

[1] https://shemesh.larc.nasa.gov/fm/fm-what.html



Logic: another thing that penguins aren't very good at.

Why formal methods?

- *non-deterministic nature*

Unlike sequential programs, the execution path of an instance of running a concurrent program cannot be determined beforehand

Why formal methods?

- Writing specification forces one to think clearly
- Excellent (unambiguous) documentation
- Find unidentified bugs

How can something be verified?

- Describe the system in a modelling language
- Write the *specification* i.e, expected property(ies) in a logic
- Verify whether the system’s description satisfies the properties

Proof-based

- System is a set of formulas Γ in a logic
- Specification is a formula φ in the same logic
- Verification consists of trying to find a proof such that $\Gamma \vdash \varphi$

$\Gamma \vdash \varphi$ means *φ can be proved from Γ*

Model Checking

- System is a Model M for an appropriate logic
- Specification is a formula φ
- Verification consist of checking whether $M \models \varphi$

$M \models \varphi$ means *M satisfies φ* or *φ is valid in M*

Proof-based vs Model Checking

- Both methods are equivalent in principle...

… but why?!

Soundness and Completeness

- Soundness: $\Gamma \vdash \varphi$ implies $\Gamma \vDash \varphi$
- Completness: $\Gamma \vDash \varphi$ implies $\Gamma \vdash \varphi$

Logic proof systems are often sound and complete, hence: $\Gamma \vdash \varphi$ iff $\Gamma \vDash \varphi$

Model checking is *simpler*

- $\Gamma \models \varphi$ means φ is valid for all possible models!
- We are only concerned with a specific model M which satisfied φ

Proof-based vs Model Checking

- Proof based
  - Requires manual intervention / semi-automatic
  - Infinite domains
  - Theorem provers, Proof assistants (Agda, Isabelle)
- Model Checking
  - Automatic
  - Finite domains
  - Model Checkers (TLC, SPIN)

What is a Model?

A *model* describes the domain of concern.

In *temporal logic*, the description of a domain is a transition system.

Model Checking

Model checking is simply finding a counterexample for φ by exhaustive search of the domain.

More specifically, finding a *counter trace* of execution

How can something be verified?

- Describe the system in a modelling language
- Write the *specification* i.e, expected property(ies) in a logic
- Verify whether the system’s description satisfies the properties

What Language?

- PlusCal (or TLA+ itself)
- PROMELA (**PRO**cess **ME**ta **LA**nguage)
- other?

What Logic?

- Propositional logic?
- Predicate logic?
- Temporal Logic!
  - LTL - A linear model of time
  - CTL - A branched model of time

LTL

$$\varphi ::= \mid \bot \mid p \mid (! \varphi) \mid (\varphi \wedge \varphi) \mid (\varphi \vee \varphi) \mid (\varphi \rightarrow \varphi)$$

$$\mid (X \varphi) \mid (F \varphi) \mid (G \varphi)$$

SPIN

Simple **P**romela **IN**terpreter

- Simulator
- Model checker
- Interpreter

Unsafe Counter - demo

- Source

Safe counter - demo

- Source