Formal Methods in the Real World

Rafael Castro - rafaelcgs10.github.io/coq

22/07/2020

Softwares causam desastres

- Therac-25: máquina de radioterapia. 6 pessoas morreram.
- Ariane 5: Foguete fez curso errado e se auto-destruiu. Colocaram um valore de 64bits num variável de 16bits.
- Knight Capital Group: perdeu 460 milhões de dólares no mercado de ações americano.

Métodos formais

- Engenharia de Software
- Confiança de software:
  1. Especificação: modelos/lógicas/tipos
  2. Desenvolvimento: baseado na espeficiação
  3. Verificação: mostrar que o desenvolvimento atende a especificação

O que é um software correto?

- Navegador
- Software de usina nuclear
- Compilador
- Linguagem de Programação (uma linguagem também segue uma especificação formal)

Confiança em software

- Testes vs Verficiação formal
- Para o que cada um serve?
- "Testes não provam a ausência de bugs, somente a presença deles"
- Provas substituem testes?

Verificações automáticas

- Verificação automática por sistemas de tipos
- Analisadores estáticos (linters) - Rubocop não é linter
- Rust: Tipos lineares, sem mutabilidade, facilita a concorrência
- Haskell: Sistema de tipos DM, sem efeitos colaterais

Model Checking

- (não manjo)
- Redes de Petri, Cadeias de Markov, Sistema de Eventos (modelos finitos)
- Muito utilizado na verificação de hardware
- Definina em modelo matemático para especificar o comportamento de alto nível
- O model check realiza verificações sobre o modelo e identifica problemas/estados indesejados
- TLA+

Linguagens com Sistemas de Tipos Dependentes

- O que são sistemas de tipos dependentes
- Idris, Agda, Epigram, F*

```
app : Vect n a -> Vect m a -> Vect (n + m) a
app Nil       ys = ys
app (x :: xs) ys = x :: app xs ys
```

Assistentes de Provas

- São ferramentas que permitem o desenvolvimento de provas matemáticas
- Programas verificados podem ser extraídos
- Coq, Isabelle, Twelf

```
Theorem plus_id_example : forall n m:nat,
  n = m -> n + n = m + m.
Proof.
  intros n m.
  intros H.
  rewrite -> H.
  reflexivity.
Qed.
```

Compcert

- Compilador de C sem bugs!
- Desenvolvido em Coq
- `http://compcert.inria.fr/`
- `https://www.absint.com/compcert/index.htm`

Sel4

- Micro-Kernel sem (certas) falhas de segurança
- Garante o isolamento entre aplicações do sistema
- `https://sel4.systems/`
- Provas em Isabelle

Linguagens para BlockChain/SmartContracts

- É importante garantir que o programa compute dentro de uma expectativa de tempo
- Garantir questões de segurança: o contrato não pode ser quebrado

Projeto DeepSpec

- `https://deepspec.org/main`