Introduction aux Machines Virtuelles avec VMKit

Ingénieur: Harris Bakiras, Responsables: Gaël Thomas, Gilles Müller EQUIPE REGAL LIP6 – INRIA (Paris/France)



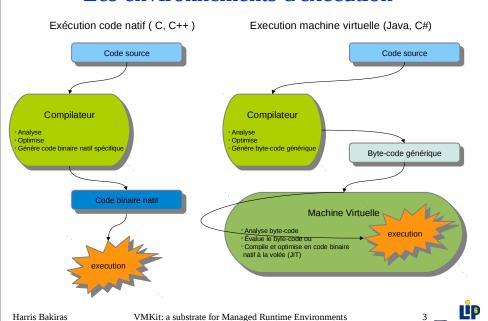
VMKit un substrat de machine virtuelle

Harris Bakiras

VMKit: a substrate for Managed Runtime Environments



Les environnements d'exécution



VMKit: a substrate for Managed Runtime Environments

Machine Virtuelle

Les machines virtuelles

vérificateur de byte-code

Compilateur à la volée (JIT) Interpréteur

Gestionnaire Threads Allocateur mémoire

Ramasse miette (GC)

Problèmes

Développement extrêmement long à réaliser!

Comment tester une idée avec différents langages?

Comment implémenter une nouvelle VM efficace pour de nouveaux langages ?

Comment étendre rapidement des langages existant?

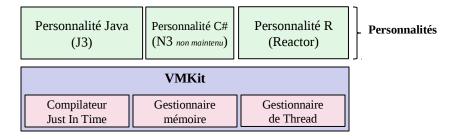
Harris Bakiras

VMKit : un substrat de machine virtuelle

But de VMKit : aider à expérimenter dans les VM

Objectif: factoriser les composants communs des VM

- Compilateur Just In Time : génération de code natif à la volée
- Gestionnaire mémoire : alloue et collecte automatiquement la mémoire libre
- Gestionnaire de Thread : créé et synchronise les threads



Harris Bakiras

VMKit: a substrate for Managed Runtime Environments

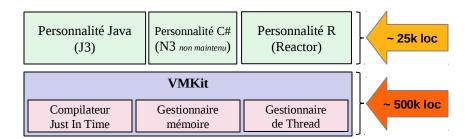


VMKit : un substrat de machine virtuelle

But de VMKit : aider à expérimenter dans les VM

Objectif: factoriser les composants communs des VM

- Compilateur Just In Time : génération de code natif à la volée
- Gestionnaire mémoire : alloue et collecte automatiquement la mémoire libre
- Gestionnaire de Thread : créé et synchronise les threads



Harris Bakiras

VMKit: a substrate for Managed Runtime Environments

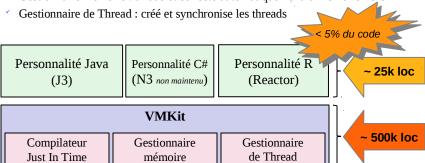


VMKit: un substrat de machine virtuelle

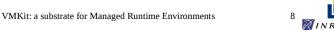
But de VMKit : aider à expérimenter dans les VM

Objectif: factoriser les composants communs des VM

- Compilateur Just In Time : génération de code natif à la volée
- Gestionnaire mémoire : alloue et collecte automatiquement la mémoire libre



VMKit d'un point de vue technique





Implementation de VMKit

Choix d'implémentation: repose sur des composants externes

✓ Compilateur à la volée (JIT) : <u>LLVM</u> [Lattner & Adve – CGO'04]

✓ Gestionnaire mémoire : MMTk [Blackburn et Al. – ICSE'04]

✓ Gestionnaire de Thread : Posix

VMKit = glue entre les différents composants

• Entre JIT-C et gestionnaire mémoire = GC précis

• Entre le gestionnaire de Thread et le gestionnaire mémoire = GC multi-threadé

Harris Bakiras

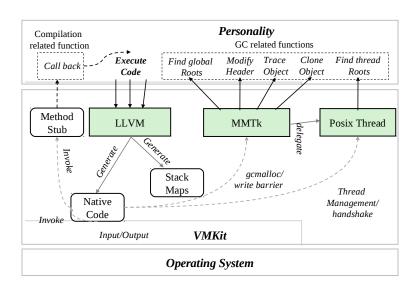
Harris Bakiras

VMKit: a substrate for Managed Runtime Environments



VM Minimal (ToyVM) pour le tutoriel

Architecture détaillée



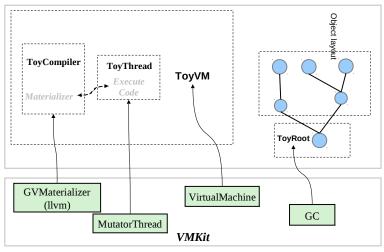
Harris Bakiras

VMKit: a substrate for Managed Runtime Environments

O ZINRI.

Architecture de la ToyVM

Personality Skeleton



Harris Bakiras

VMKit: a substrate for Managed Runtime Environments



Architecture de la ToyVM

Personality Skeleton ToyCompiler ToyRoot GVMaterializer VirtualMachine (llvm) GC MutatorThread **VMKit**

Harris Bakiras

VMKit: a substrate for Managed Runtime Environments

ToyVM et ToyThread

ToyVM ← VirtualMachine

- ✓ Point d'entrée du Garbage collector
- ✓ Backtrace (parcours de la pile d'exécution)
- ✓ Traçage des variables globales
- ✓ Gestion des exceptions

$ToyThread \leftarrow MutatorThread \leftarrow Thread$

- Rendez-vous du garbage collector
- √ Scan de la pile d'exécution lors du GC
- ✓ Traçage des variables locales
- ✓ Méthode principale d'exécution

Harris Bakiras

VMKit: a substrate for Managed Runtime Environments

Architecture de la ToyVM

Personality Skeleton ToyCompiler ToyThread ToyVM ToyRoot GVMaterializer VirtualMachine (llvm) GC MutatorThread **VMKit**

Architecture de la ToyVM

Personality Skeleton ToyCompiler ToyThread ToyVM GVMaterializer VirtualMachine (llvm) GC MutatorThread **VMKit**



ToyRoot (tag)

ToyRoot ← vmkit::gc

- Tagger les objets (stack maps)

```
[ ...... ]

ToyRoot* F (ToyRoot* param) {
    TOY_PARAM(param);
    TOY_VAR(ToyRoot, val);
    [init val ...]
    val = g(param, val);
    return val;
}

ToyRoot* G (ToyRoot* a, ToyRoot* b){
    TOY_PARAM(b);
    TOY_PARAM(b);
    TOY_PARAM(b);
    TOY_VARAM(b);
    res = a.doSomething(b);
    return res;
}

[ ....... ]
```

Harris Bakiras

VMKit: a substrate for Managed Runtime Environments

17 ZINRIA

ToyRoot (tag)

ToyRoot ← vmkit::gc

- Tagger les objets (stack maps)

```
ToyRoot* F (ToyRoot* param) {
TOY_PARAM(param);
TOY_PARAM(param);
Init val...]
val = g(param, val);
return val;
}

ToyRoot* G (ToyRoot* a, ToyRoot* b){
Toy_PARAM(a);
TOY_PARAM(b);
TOY_PARAM(b);
TOY_PARAM(b);
resurn res;
}

[......]
```

Harris Bakiras

VMKit: a substrate for Managed Runtime Environments



ToyRoot (tag)

$ToyRoot \leftarrow vmkit::gc$

- Tagger les objets (stack maps) Call stack parameters ToyRoot* F (ToyRoot* param) { TOY_PARAM(param); TOY_VAR(ToyRoot, val) [init val ...] Local val = g(param, val); return val; Stack ToyRoot* G (ToyRoot* a, ToyRoot* b){
TOY PARAM(a); parameters Maps TOY PARAM(b); TOY VAR(ToyRoot, res) ; res = a.doSomething(b); Local return res : vars [......]

(11 8)

ToyRoot (header gc)

ToyRoot ← vmkit::gc

- Tagger les objets (stack maps)
- header GC
 - Bits de marquage
- Bits de lock (J3)
- Bits de hash
- Bits personnalisables



Harris Bakiras

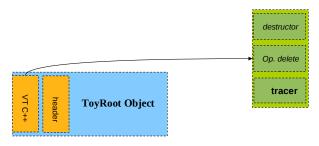
VMKit: a substrate for Managed Runtime Environments



ToyRoot (header gc)

ToyRoot ← vmkit::gc

- Tagger les objets (stack maps)
- header GC
 - Bits de marquage
- Bits de lock (J3)
- Bits de hash
- Bits personnalisables
- Méthode de Traçage de l'objet



Harris Bakiras

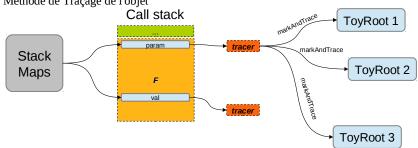
VMKit: a substrate for Managed Runtime Environments

ToyRoot (tracer)

ToyRoot ← vmkit::gc

- Tagger les objets (stack maps)
- header GC
 - Bits de marquage
- Bits de lock (J3)
- Bits de hash
- Bits personnalisables

- Méthode de Traçage de l'objet



Harris Bakiras

VMKit: a substrate for Managed Runtime Environments

ToyRoot (tracer)

ToyRoot ← vmkit::gc

- Tagger les objets (stack maps)
- header GC
 - Bits de marquage
- Bits de lock (J3)
- Bits de hash
- Bits personnalisables
- Méthode de Traçage de l'objet
- Allocateur GC

Harris Bakiras

- Surcharge operator new
- Appel à *new* interdit pour gc (paramètre opaque)

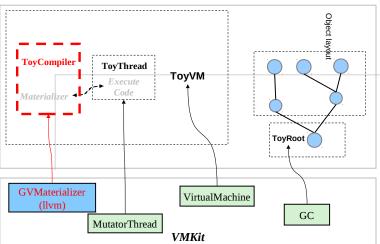
Architecture de la ToyVM

Personality Skeleton ToyCompiler ToyThread ToyVM ToyRoot GVMaterializer VirtualMachine (llvm) GC MutatorThread **VMKit**



Architecture de la ToyVM

Personality Skeleton



Harris Bakiras

VMKit: a substrate for Managed Runtime Environments



ToyCompiler (load IR)

ToyCompiler ← GVMaterializer

Récupération du code IR (représentation intermédiaire LLVM)







Harris Bakiras

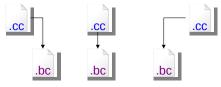
VMKit: a substrate for Managed Runtime Environments



ToyCompiler (load IR)

ToyCompiler ← GVMaterializer

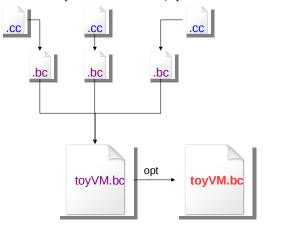
Récupération du code IR (représentation intermédiaire LLVM)



ToyCompiler (load IR)

ToyCompiler ← GVMaterializer

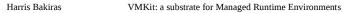
Récupération du code IR (représentation intermédiaire LLVM)



Harris Bakiras

VMKit: a substrate for Managed Runtime Environments

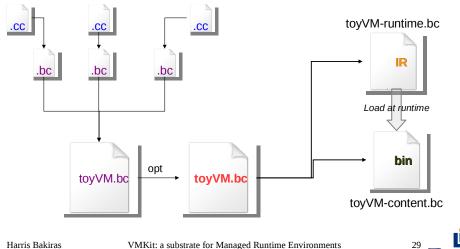




ToyCompiler (load IR)

ToyCompiler ← GVMaterializer

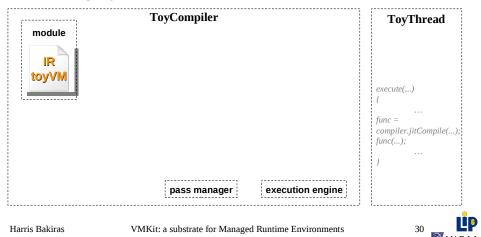
Récupération du code IR (représentation intermédiaire LLVM)



ToyCompiler (generate IR)

ToyCompiler ← GVMaterializer

- Récupération du code IR (représentation intermédiaire LLVM)
- JIT compiling

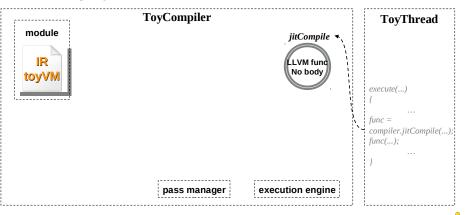


ToyCompiler (generate IR)

ToyCompiler ← GVMaterializer

- Récupération du code IR (représentation intermédiaire LLVM)
- JIT compiling

Harris Bakiras

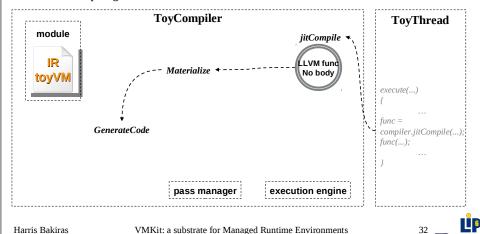


VMKit: a substrate for Managed Runtime Environments

ToyCompiler (generate IR)

ToyCompiler ← GVMaterializer

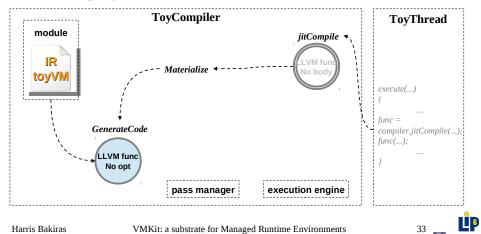
- * Récupération du code IR (représentation intermédiaire LLVM)
- JIT compiling



ToyCompiler (generate IR)

ToyCompiler ← GVMaterializer

- Récupération du code IR (représentation intermédiaire LLVM)
- JIT compiling

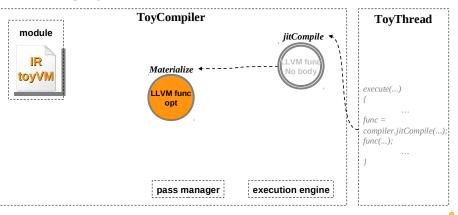


ToyCompiler (optimize IR)

ToyCompiler ← GVMaterializer

- Récupération du code IR (représentation intermédiaire LLVM)
- ✓ JIT compiling

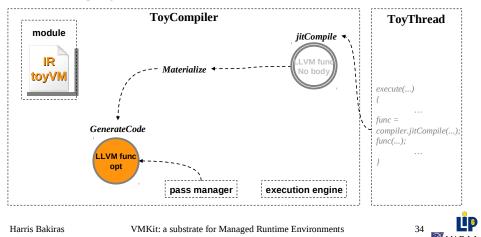
Harris Bakiras



ToyCompiler (optimize IR)

ToyCompiler ← GVMaterializer

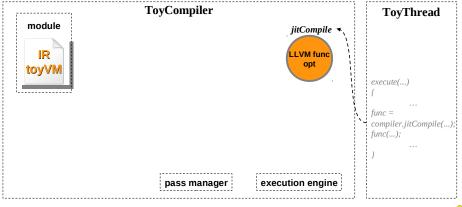
- Récupération du code IR (représentation intermédiaire LLVM)
- JIT compiling



ToyCompiler (optimize IR)

ToyCompiler ← GVMaterializer

- * Récupération du code IR (représentation intermédiaire LLVM)
- JIT compiling



Harris Bakiras

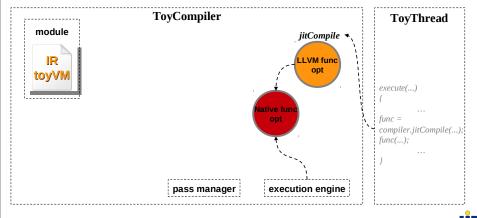
VMKit: a substrate for Managed Runtime Environments

36 N.N.P.

ToyCompiler (IR to native)

ToyCompiler ← GVMaterializer

- Récupération du code IR (représentation intermédiaire LLVM)
- JIT compiling



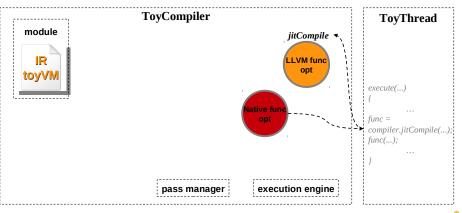
ToyCompiler (IR to native)

VMKit: a substrate for Managed Runtime Environments

ToyCompiler ← GVMaterializer

- Récupération du code IR (représentation intermédiaire LLVM)
- ✓ JIT compiling

Harris Bakiras



ToyCompiler (GC Infos)

ToyCompiler ← GVMaterializer

- Récupération du code IR (représentation intermédiaire LLVM)

Harris Bakiras

VMKit: a substrate for Managed Runtime Environments

88 ZINRIA

ToyCompiler

ToyCompiler ← GVMaterializer

- Récupération du code IR (représentation intermédiaire LLVM)
- JIT compiling
 - Generation d'IR
 - · Optimisation de l'IR
 - · Conversion de l'IR vers code natif
- Transmission des StackMaps (information variables locales) au GC



Questions?

Harris Bakiras VMKit: a substrate for Managed Runtime Environments



A vos claviers!

