GraalVM™

High Performance. Cloud Native. Polyglot.



Companies that successfully use different GraalVM technologies in their products and services

facebook
standard chartered
Alibaba Group
ORACLE Cloud Infrastructure
Odnoklassniki
NVIDIA.
ORACLE NETSUITE
POLITIE
Goldman Sachs

Copyright © 2020, Oracle and/or its affiliates | Confidential: Internal

Naučno stručna saradnja: Matematički fakultet i istraživačko-razvojni centar Oracle Labs

Oracle Labs

dr Vojin Jovanović, Senior Research Manager

Matematički fakultet

prof. dr Milena Vujošević Janičić

Doktorandi

Milan Čugurović

Marjana Gligorijević

Ivan Ristović

Strahinja Stanojević

Copyright © 2021, Oracle and/or its affiliates | Confidential: Internal

Machine-Learning-Based Compiler-Parameter and Profile Prediction

- Zvanični početak projekta: 1. jun 2021. godine

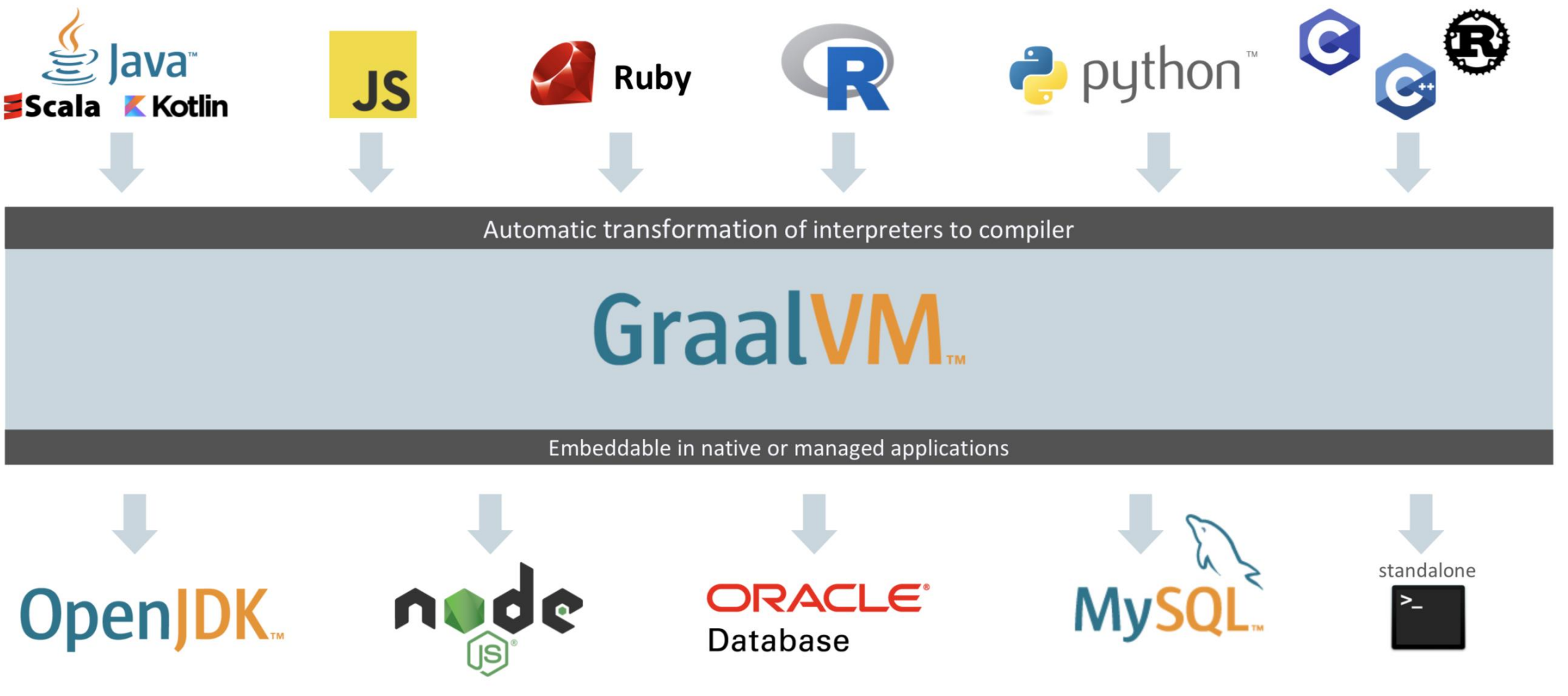
- Vojin Jovanović
  - MG, ETF, MI SANU, EPFL Švajcarska
  - Doktorat 2015. godine, mentor Martin Odersky
  - Oracle Labs (Senior Research Manager)
  - *Izvršavanje programa treba da bude efikasno bez obzira na konciznost i apstraktnost u njegovom opisivanju*

Copyright © 2021, Oracle and/or its affiliates | Confidential: Internal

Pregled izlaganja

- Uvod u GraalVM projekat – **Vojin Jovanović**
- Tekući potprojekti
  - Primene mašinskog učenja u okviru GraalVM-a
    - Predviđanje profila - **Milan Čugurović**
    - Predviđanje parametara optimizacije – **Milena Vujošević Janičić**
  - Statička analiza i poliglotizam
    - Praktični aspekti statičke analize – **Marjana Gligorijević**
    - Poliglotizam i GraalVM – **Ivan Ristović**
- Priključivanje projektu i otvoreni problemi
  - Prvi koraci prilikom uključivanja na projekat – **Strahinja Stanojević**
  - Otvoreni problemi – **Vojin Jovanović**
  - Zaključci i diskusija - **Milena Vujošević Janičić**

Copyright © 2021, Oracle and/or its affiliates | Confidential: Internal



Copyright © 2021, Oracle and/or its affiliates | Confidential: Internal

Kompajler Graal

Napisan u Javi: jednostavan za modifikacije

Modularna arhitektura:

- Lako se dodaju nove faze, separacija od VM-a

Agresivne optimizacije zahtevaju profile po dizajnu







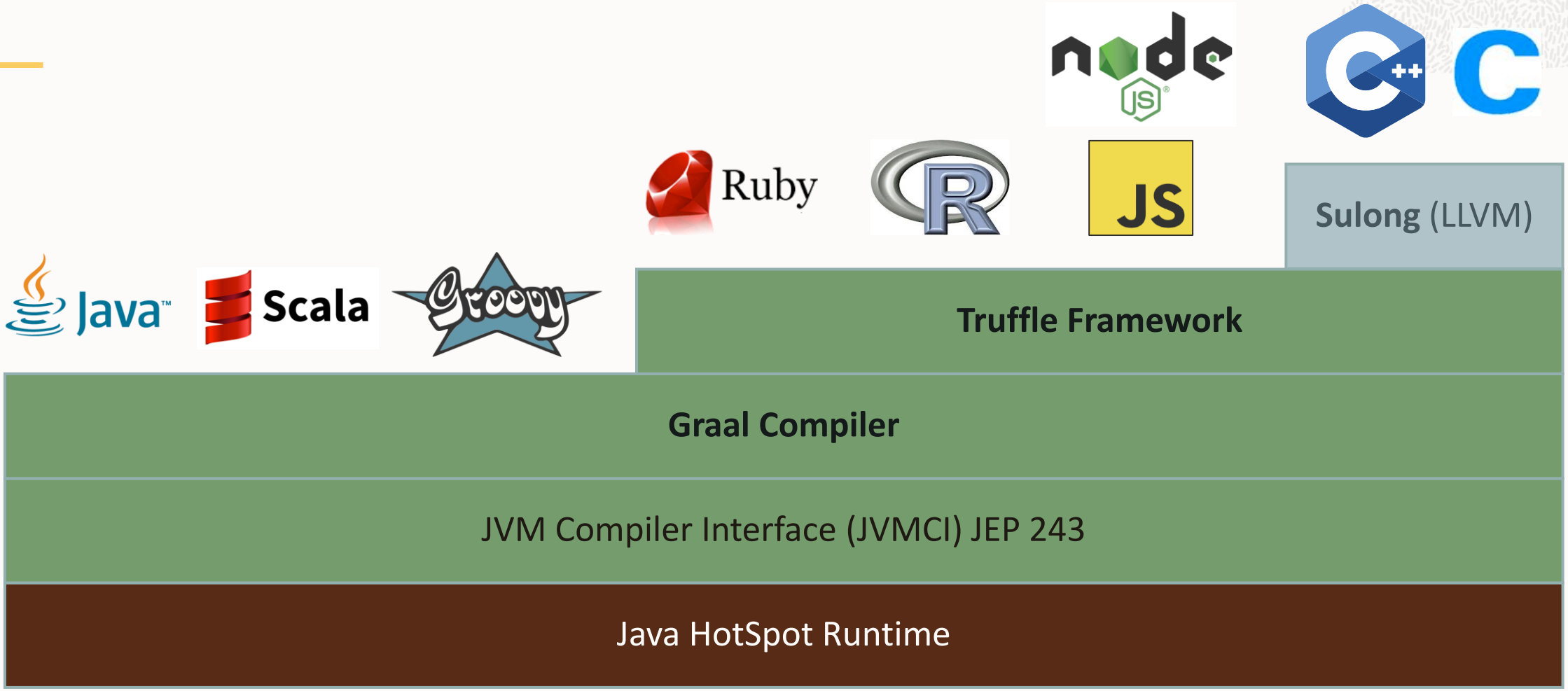
Graal Compiler
JVM Compiler Interface (JVMCI) JEP 243
Java HotSpot Runtime

```
Arrays.stream(persons)
  .filter(p -> p.getEmpl() == Employment.EMPLOYED)
  .filter(p -> p.getSalary() > 100_000)
  .mapToInt(Person::getAge)
  .filter(age -> age >= 40).average()
  .getAsDouble();
```

```
long count = 0, sum = 0;
for (int i = 0; i < persons.length; i++) {
    Person person = persons[i];
    int age = person.age;
    if (person.employment == Employment.EMPLOYED &&
            person.salary > 100_000 &&
            age >= 40) {
        sum += age;
        count += 1;
    }
}
return (double) sum / count;
```

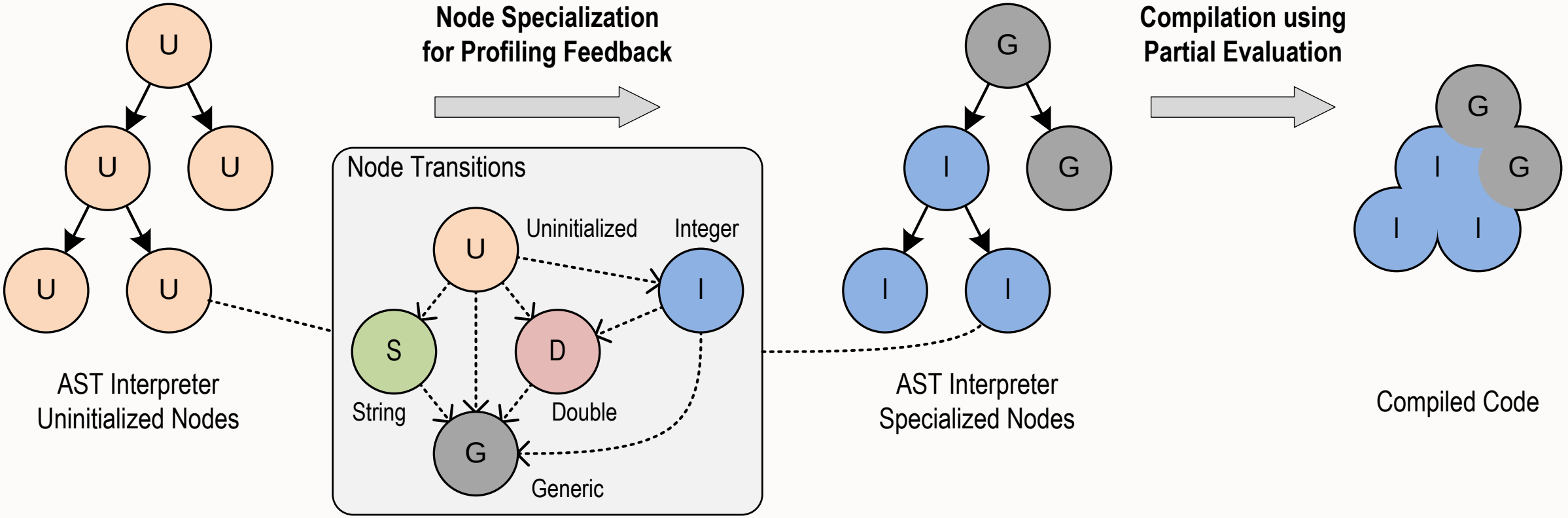

Copyright © 2021, Oracle and/or its affiliates | Confidential: Internal

Arhitektura GraalVM-a



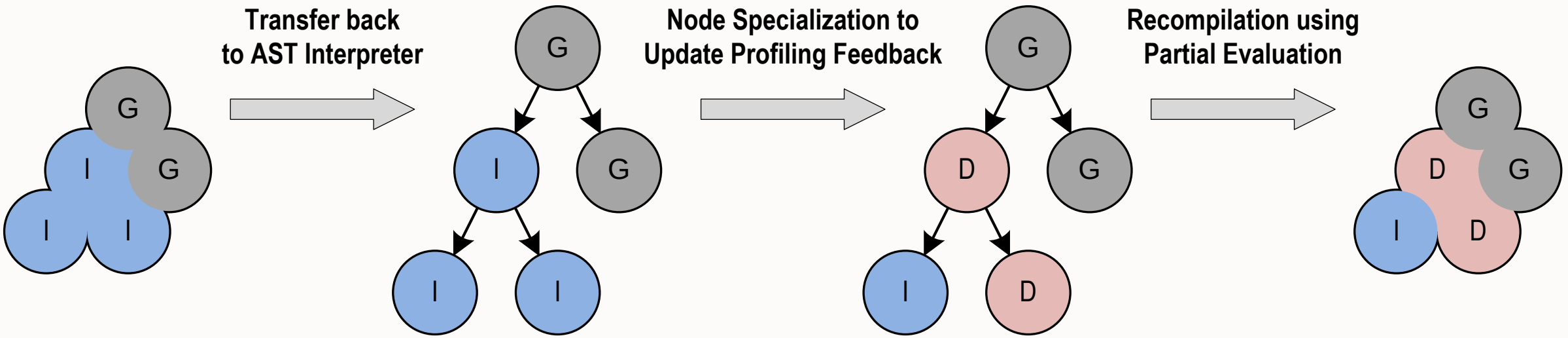
Copyright © 2020, Oracle and/or its affiliates | Confidential: Internal

Truffle: spekulacija i optimizacija



Copyright © 2020, Oracle and/or its affiliates | Confidential: Internal

… povratak u interpreter i ponovna optimizacija



Copyright © 2020, Oracle and/or its affiliates | Confidential: Internal

```
def add(a, b)
  a + b
end

def sum(n)
  i = 0
  a = 0
  while i < n
    i += 1
    a = add(a, n)
  end
  a
end
```

Ako pogledamo prevod ove petlje

Copyright © 2020, Oracle and/or its affiliates | Confidential: Internal

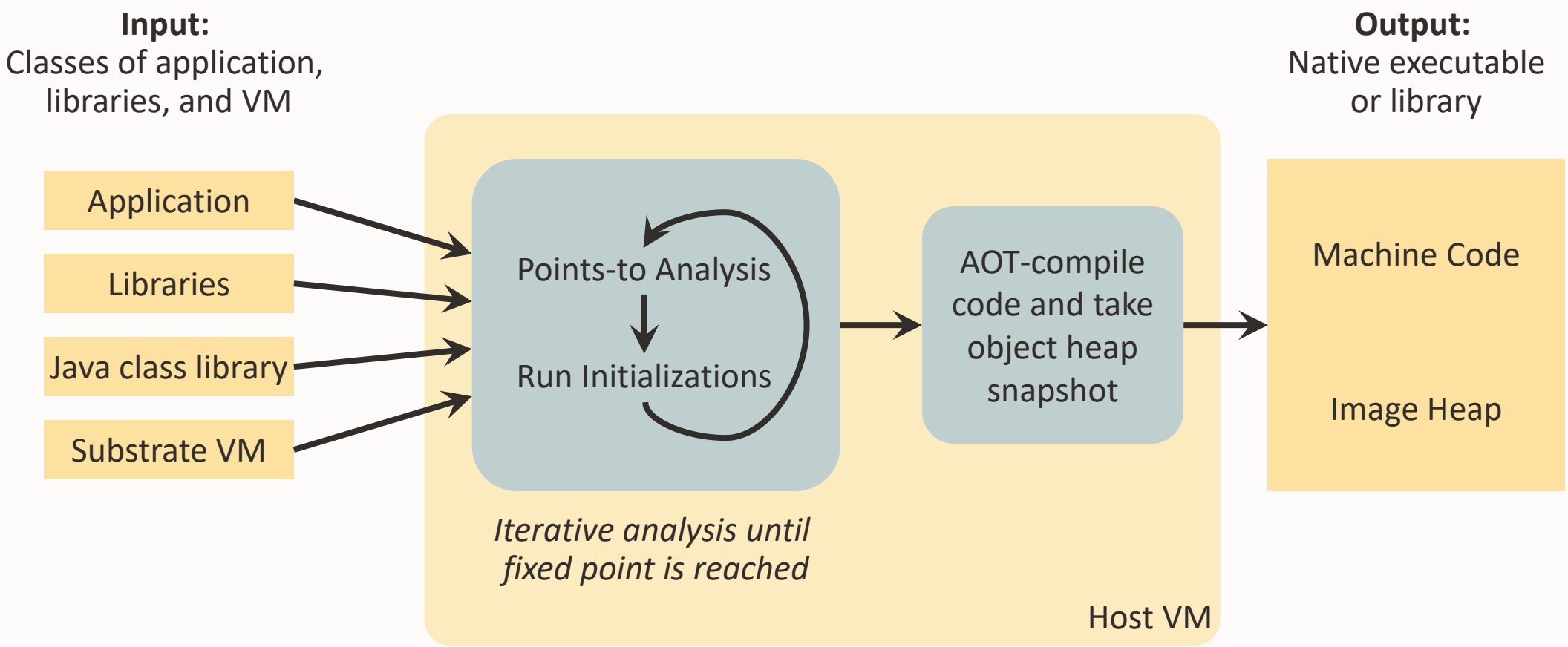
```ruby
def add(a, b)
  a + b
end

def sum(n)
  i = 0
  a = 0
  while i < n
    i += 1
    a = add(a, n)
  end
  a
end
```

```
0x0000000103a7dc70: mov    esi,edi
0x0000000103a7dc72: add    esi,r9d
0x0000000103a7dc75: jo     0x0000000103a7dda2
0x0000000103a7dc7b: inc    ecx
0x0000000103a7dc7d: mov    edi,esi
0x0000000103a7dc7f: cmp    r9d,ecx
0x0000000103a7dc82: jg     0x0000000103a7dc70
```

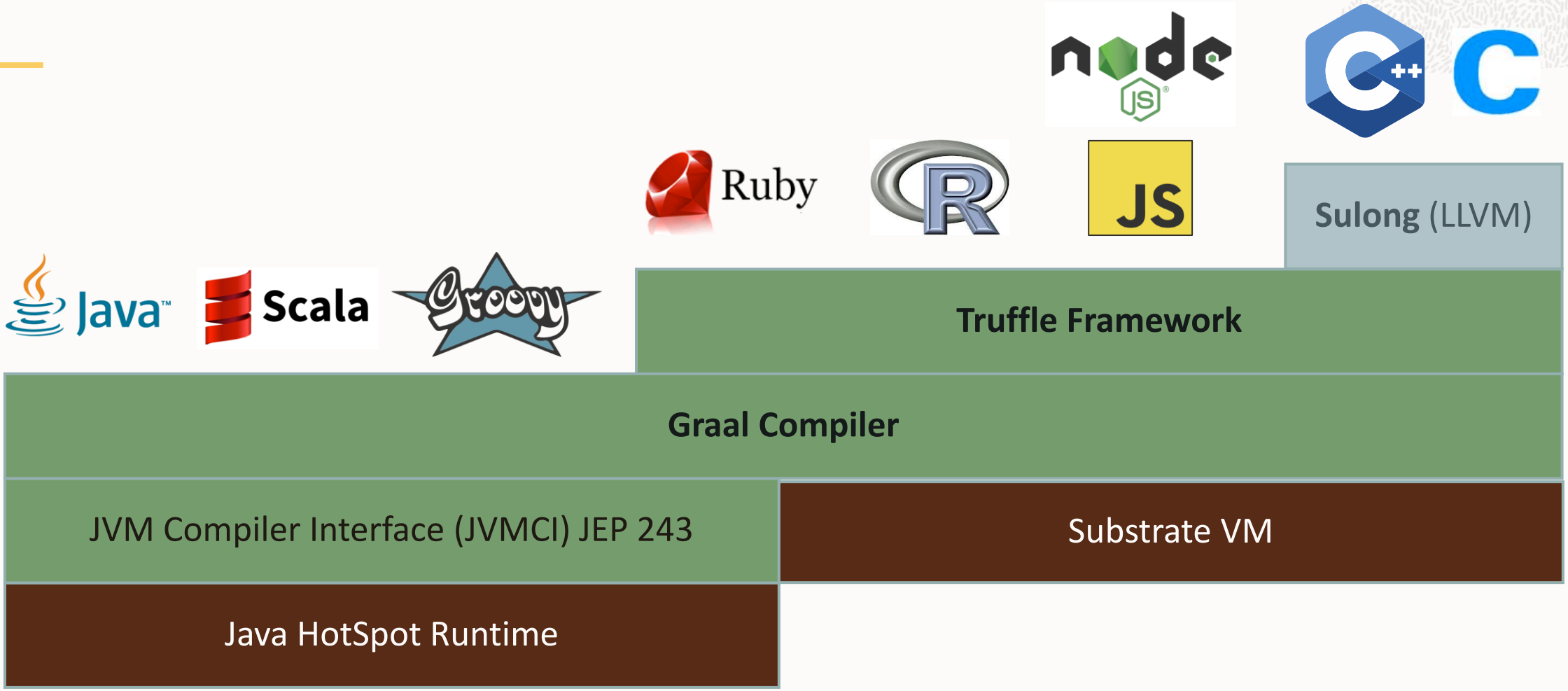

Copyright © 2020, Oracle and/or its affiliates | Confidential: Internal

GraalVM Native Image: native kompilacija i snapshotting



Copyright © 2020, Oracle and/or its affiliates | Confidential: Internal

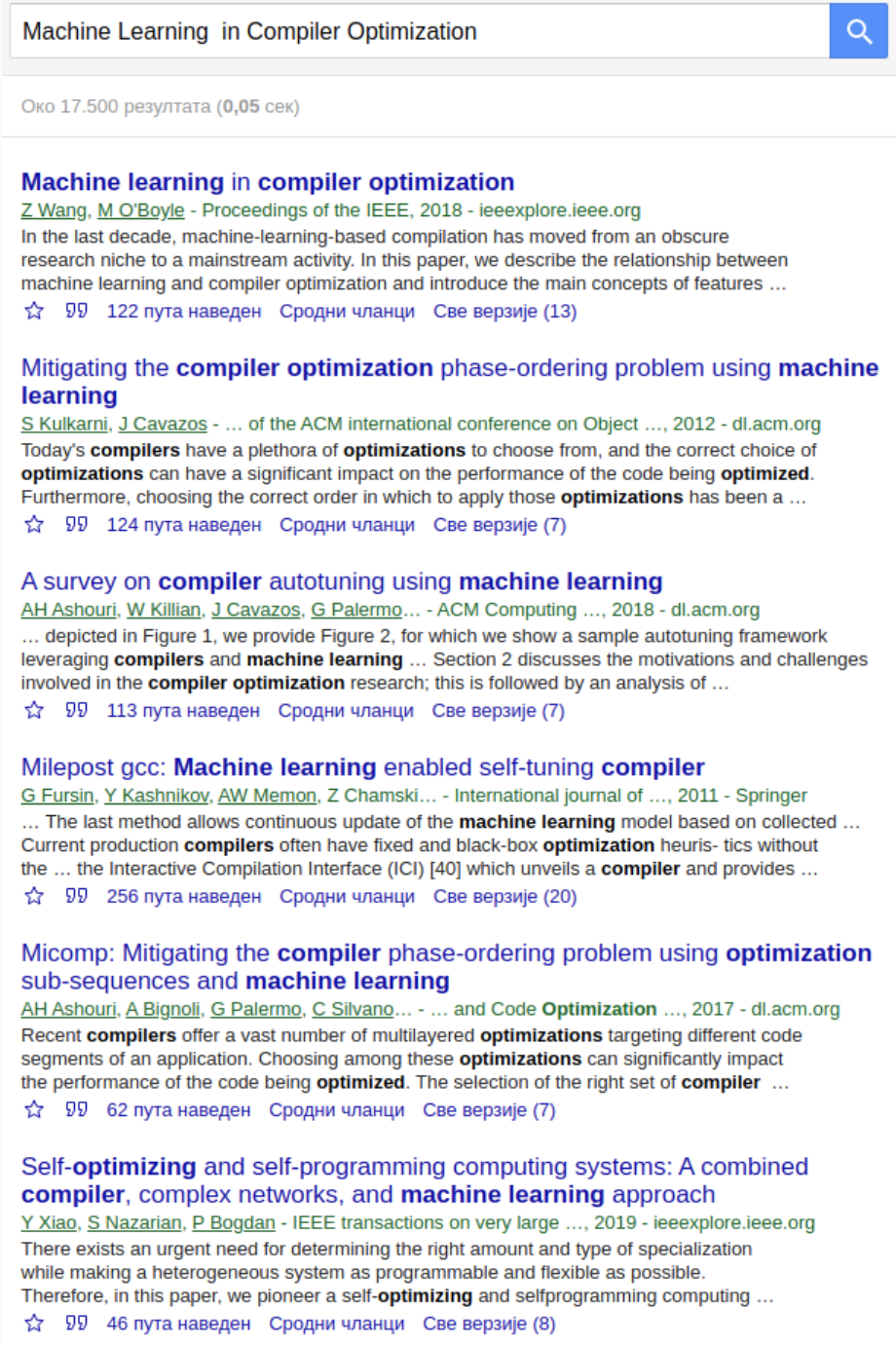
GraalVM: Arhitektura



Copyright © 2020, Oracle and/or its affiliates | Confidential: Internal

Primene mašinskog učenja u okviru GraalVM-a

- Predviđanje profila programa
- Predviđanje parametara optimizacije

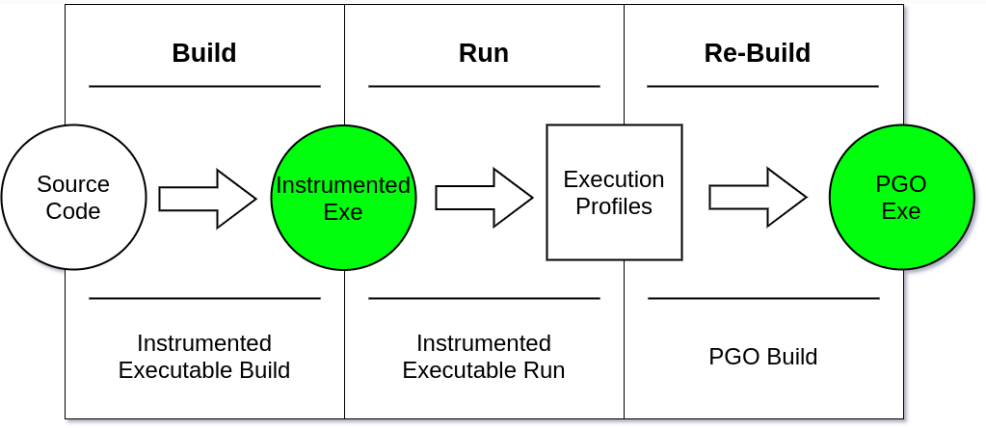


Copyright © 2021, Oracle and/or its affiliates | Confidential: Internal

Predviđanje profila metodama mašinskog učenja

Motivacija

- Optimizacije vođene profilima (PGO): inlajning, optimizacije keša, duplikacija koda, itd.
- Kvalitetni profili: instrumentalizacija/pokretanje, uobičajena upotreba, pokrivenost koda




```
public static void parallelSort(double[] a) {
    int n = a.length, p, g;
    if (n <= MIN_ARRAY_SORT_GRAN || (p = ForkJoinPool.getCommonPoolParallelism()) == 1)
        DualPivotQuicksort.sort(a, 0, n - 1, null, 0, 0);
    else
        new ArraysParallelSortHelpers.FJDouble.Sorter(null, a, new double[n], 0, n, 0, ((g = n / (p << 2))
        <= MIN_ARRAY_SORT_GRAN) ? MIN_ARRAY_SORT_GRAN : g).invoke();
}
```

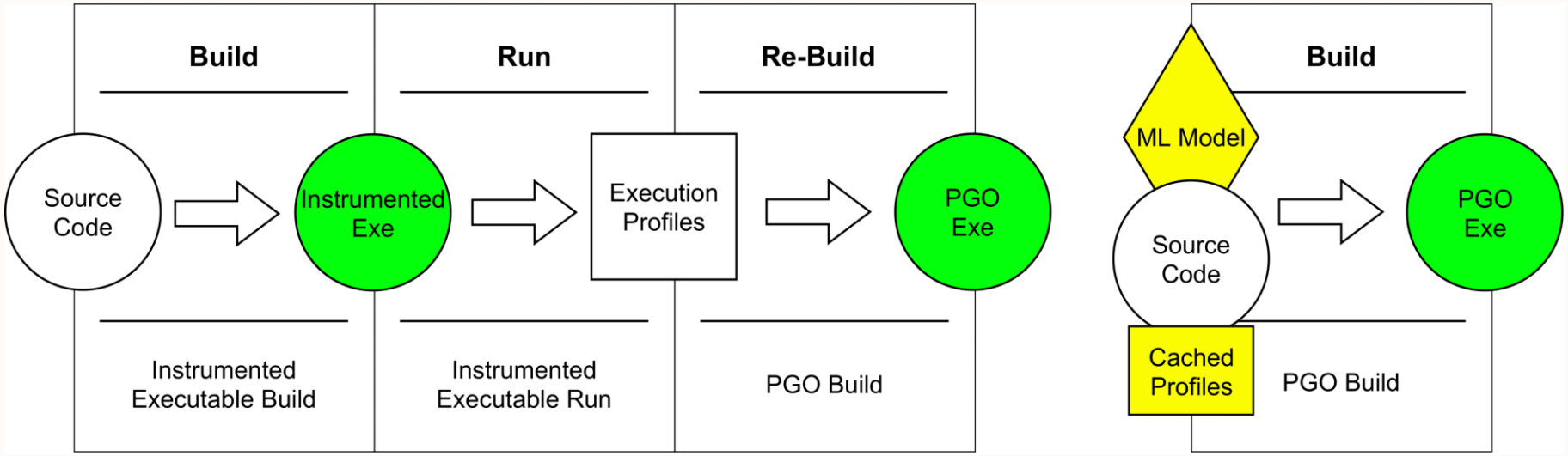
- Primer: MySQL

Copyright © 2021, Oracle and/or its affiliates | Confidential: Internal

Predviđanje profila metodama mašinskog učenja

Motivacija

- Alternativa: statička predikcija profila
  - Tradicionalne heuristike (npr. BTFNT, *bltz*, *blez* - *ne*; *bhtz*, *bgez* - da)
  - ML klasifikatori: DT, LR, DNN, KNN, RNN (klasifikacija putanja)

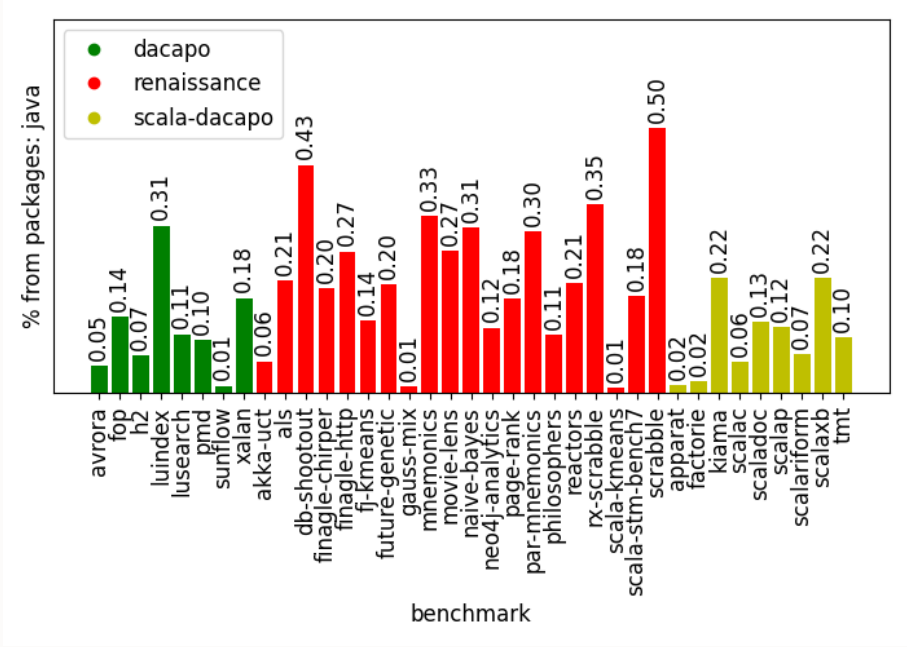


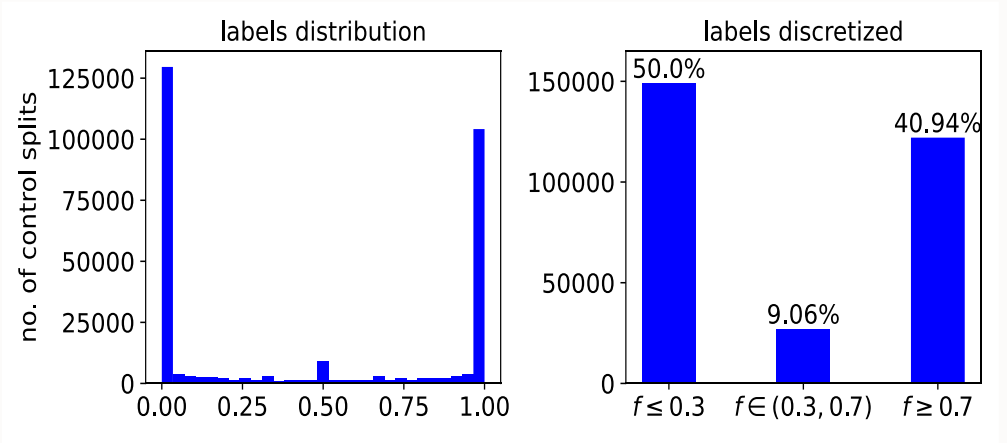
Copyright © 2021, Oracle and/or its affiliates | Confidential: Internal

Predviđanje profila metodama mašinskog učenja

Predloženo rešenje

1. Uvođenje korišćenja **keširanih profila standardne biblioteke**: prikupljanje i upotreba profila
2. Prevazilaženje problema prikupljanja profila prelaskom na **regresione modele mašinskog učenja (MLPP)**
   - Atributi koji opisuju delove koda:
     u potpunosti interpretabilni, baziran na IR, histogram+kontrola toka
   - 300k dataset: Renaissance + DaCapo{Con Scala}
   - 440 atributa:
     selekcija atributa bazirana na varijansi, skaler, PCA
   - ML regresioni modeli:
     ne vrše zaokruživanje, nema curenja informacija



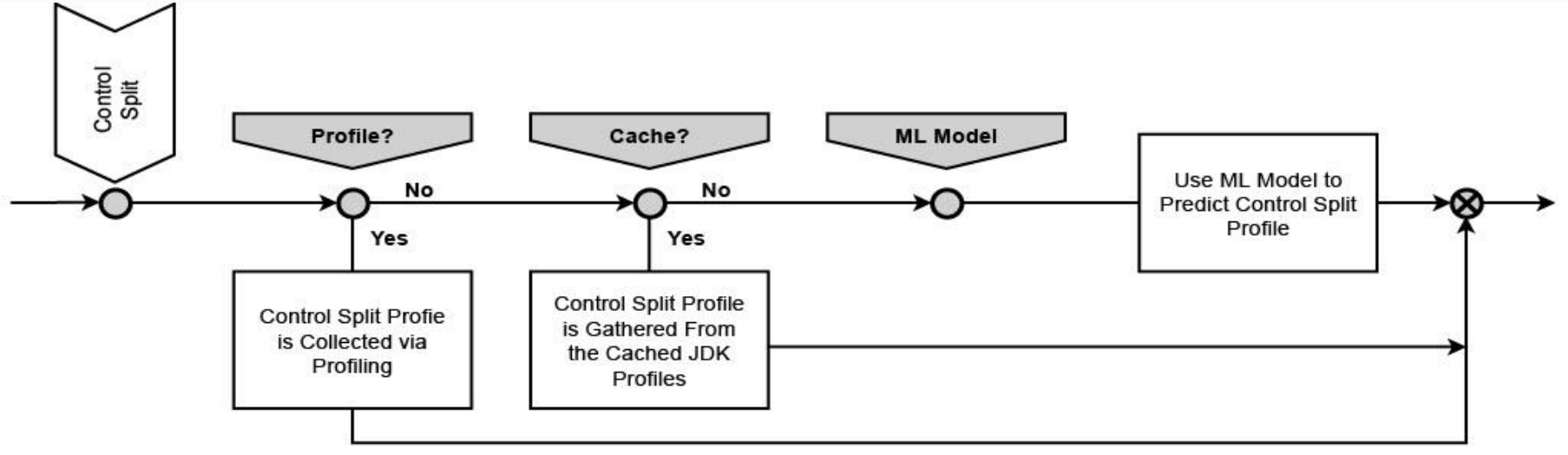


Copyright © 2021, Oracle and/or its affiliates | Confidential: Internal

Predviđanje profila metodama mašinskog učenja

Predloženo rešenje

- ML Modeli:
  - *DT*: interpretabilnost, zaključci, ~1ms
  - *DNN*: bez interpretabilnosti, 0.5M, ~4ms
- Razvoj end-to-end rešenja: JDK keš + pretrenirani ML modeli

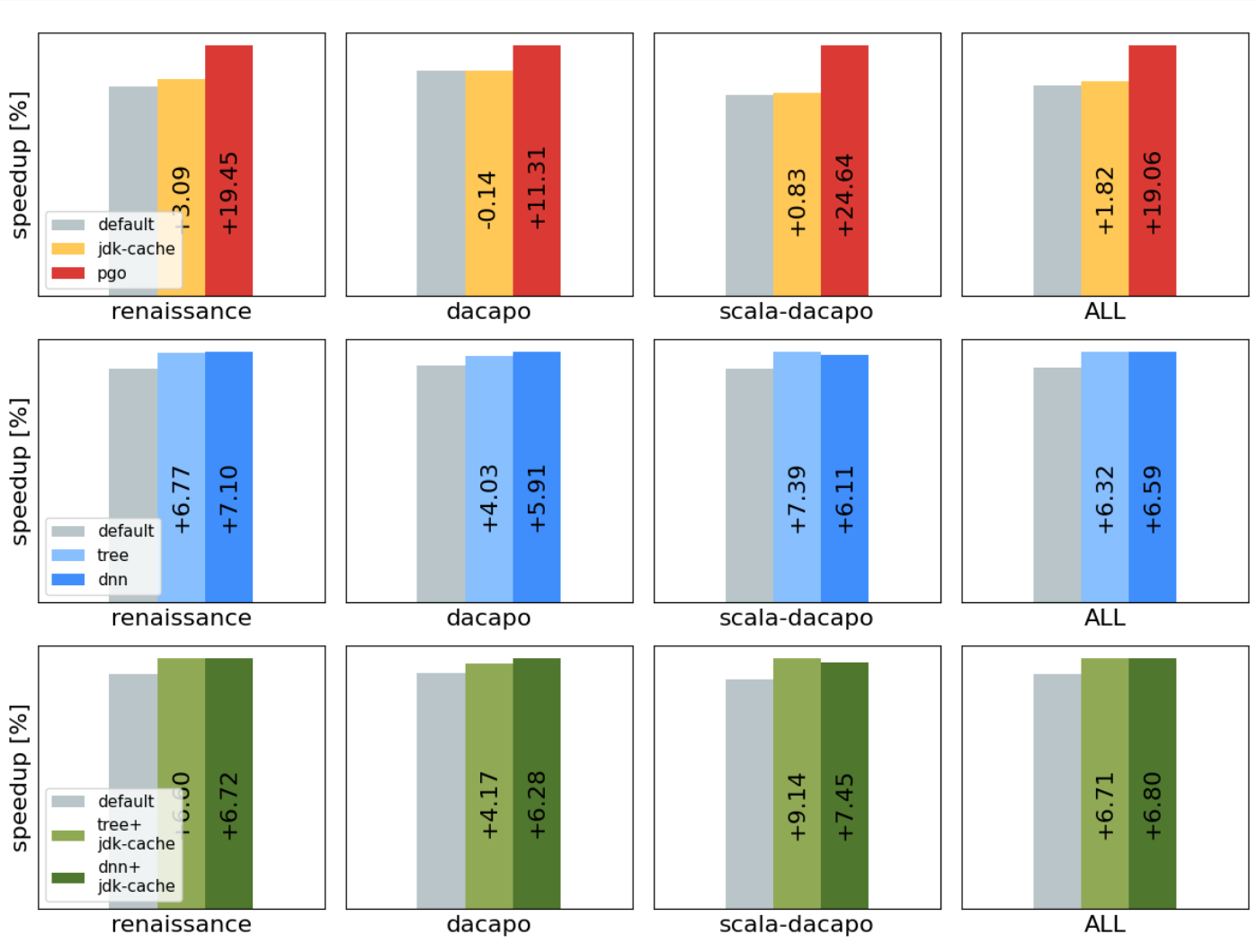


- Leave-{one, similar}-out evaluacija

Copyright © 2021, Oracle and/or its affiliates | Confidential: Internal

Predviđanje profila metodama mašinskog učenja

Rezultati



- PGO: context sensitive
- MLPP: context insensitive
- Plan:
  - hotness points
  - invoke points
- Build size: +5%
- Build time: +15%-20%

Copyright © 2021, Oracle and/or its affiliates | Confidential: Internal

Predviđanje parametara optimizacije metodama mašinskog učenja

- Kompajleri najčešće koriste fiksirane parametre za kompilaciju svih metoda
  - Propuštanje prilika za optimizaciju
  - Optimizacije u kodu koji se retko ili nikada ne izvršava

- Kako prepoznati prave parametre optimizacije i prilagoditi ih konkretnom kodu?
  - Primena tehnika mašinskog učenja: naučiti povezanost karakteristika koda i efekata optimizacija
  - Prepoznati mogućnosti optimizacije na pravim mestima, na primer izbeći duplikaciju koda tamo gde to ne dovodi do poboljšanja u fazi izvršavanja

Copyright © 2021, Oracle and/or its affiliates | Confidential: Internal

Predviđanje parametara optimizacije metodama mašinskog učenja

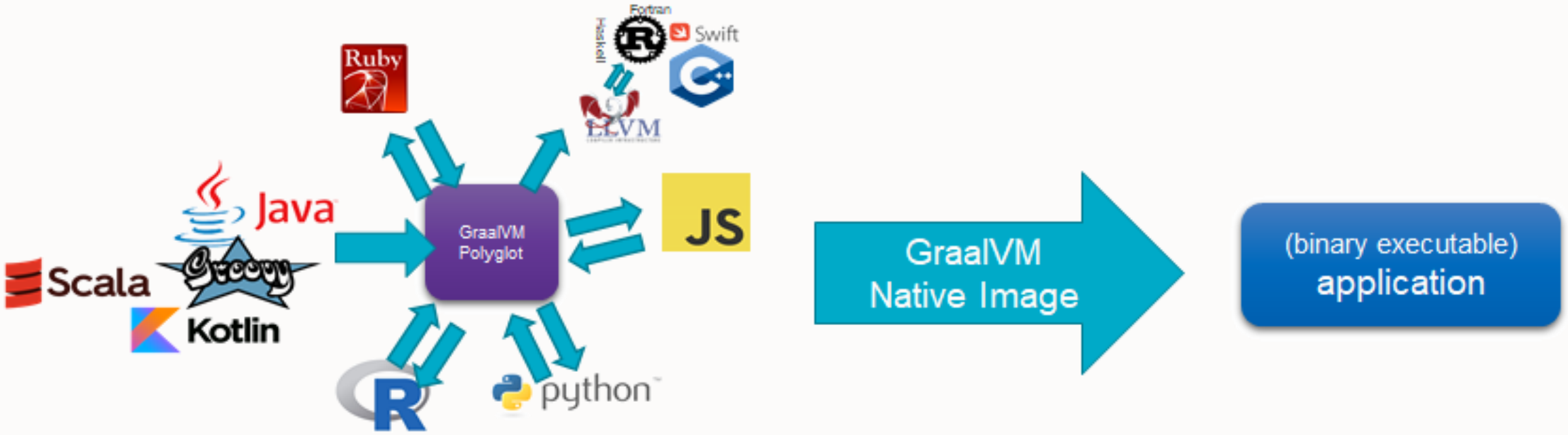
- Faze projekta
  - Prikupljanje podataka: uparivanje karakteristika koda sa rezultatima merenja za različite parametre kompilacije
  - Treniranje modela: u planu je korišćenje metode potpornih vektora i neuronske mreže
  - Implementacija: izbor najboljeg modela koji će ući u produkciju

- Očekivani rezultati
  - Bolje performanse (za GraalVM Native Image: min 3% sa PGO, min 7% bez PGO)
  - Smanjenje izvršnog fajla za 5%

Copyright © 2021, Oracle and/or its affiliates | Confidential: Internal

Statička analiza i poliglotizam

- Praktični aspekti statičke analize
- Poliglotizam i GraalVM



Copyright © 2021, Oracle and/or its affiliates | Confidential: Internal

Umetanje metoda pre statičke analize u kombinaciji sa kontekstno-senzitivnom analizom



- Statička analiza koda omogućava otkrivanje dostupnih klasa i metoda, kao i eliminaciju mnogih provera u kodu (null vrednosti, provera tipa, …) dokazivanjem njihove suvišnosti. Stoga, preciznost statičke analize je od velike važnosti za poboljšanje performansi koda.

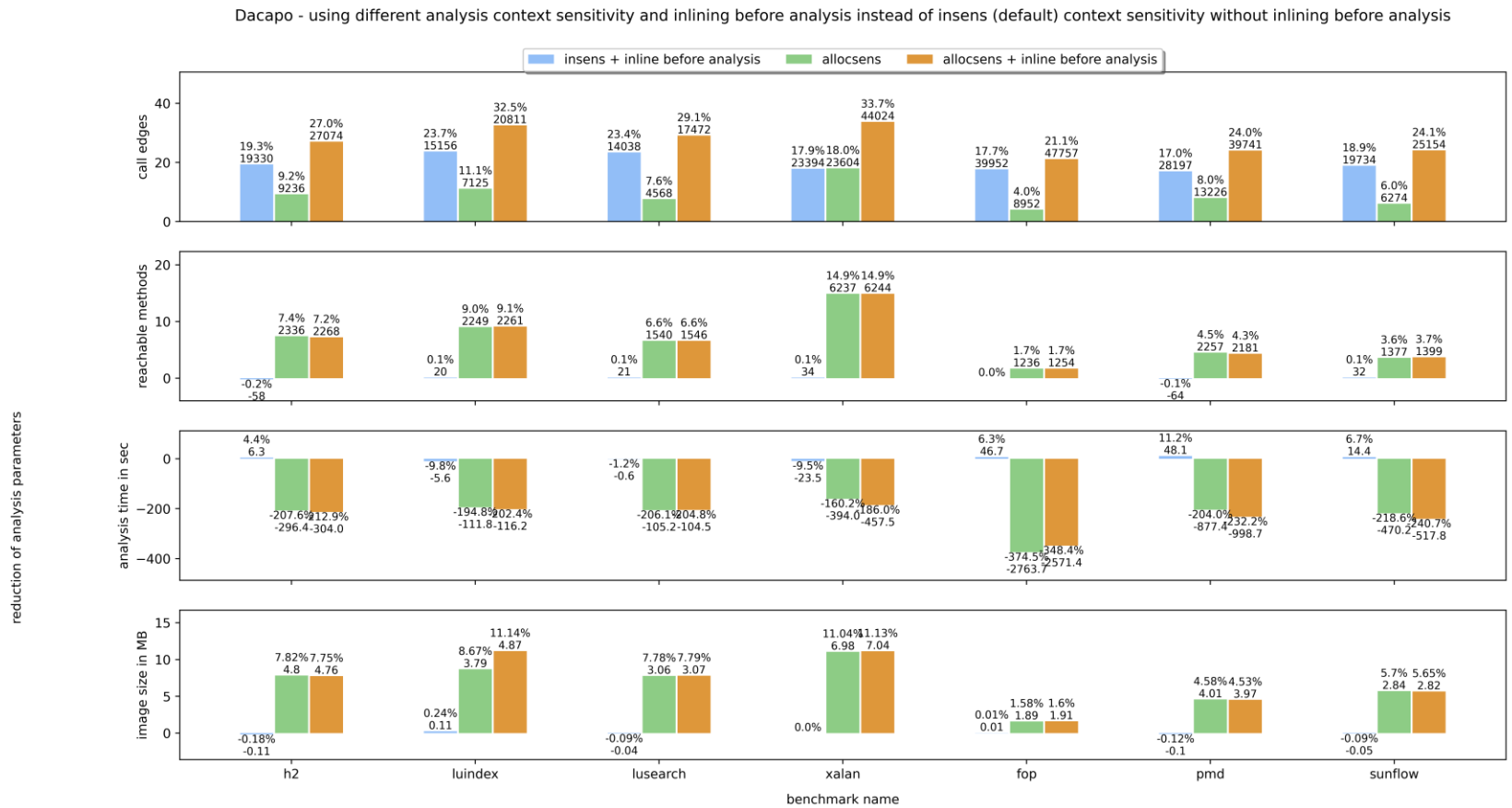
Kontekstno-senzitivna analiza	Umetanje metoda
senzitivnost poboljšava preciznost statičke analize	nestandardna upotreba pre analize
ali čini analizu znatno sporijom	nema negativan uticaj na vreme analize
veoma skupa procedura	jednostavna i jeftina procedura za male metode

umetanje malih metoda → propagacija konstanti → uklanjanje mrtvog koda pre statičke analize

- Još benefita: upotreba reflektivnih metoda bez potrebe za dodatnim konfiguracionim podacima za refleksiju (Java frejmork Spring).

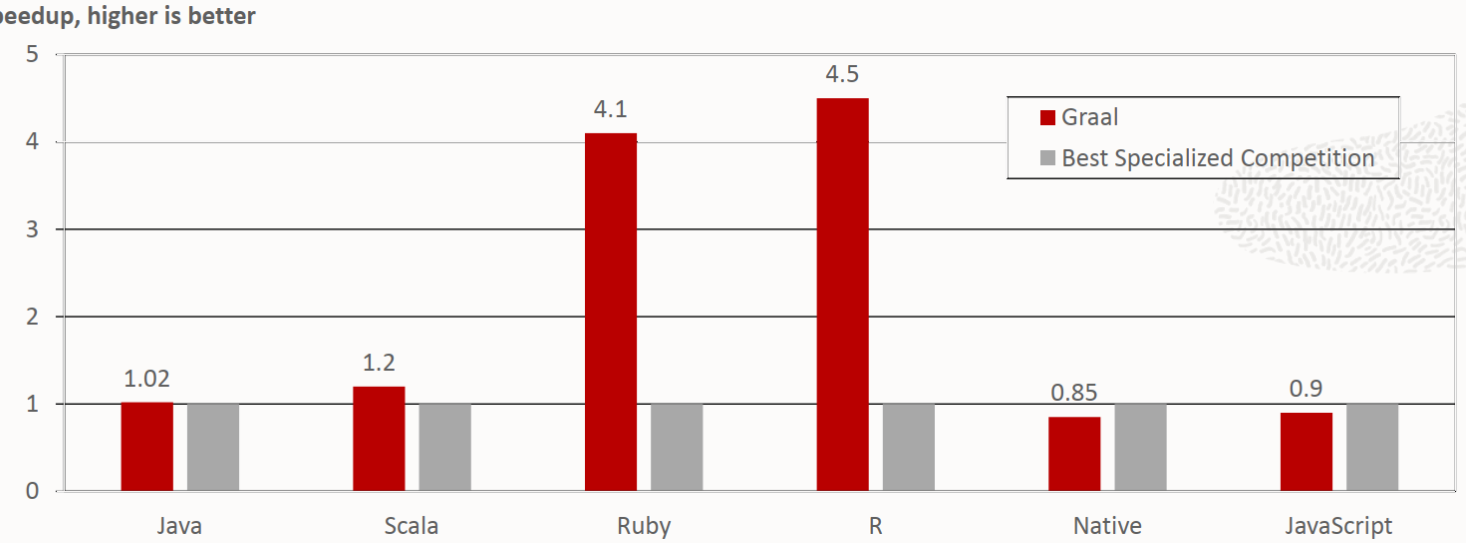
Copyright © 2021, Oracle and/or its affiliates | Confidential: Internal

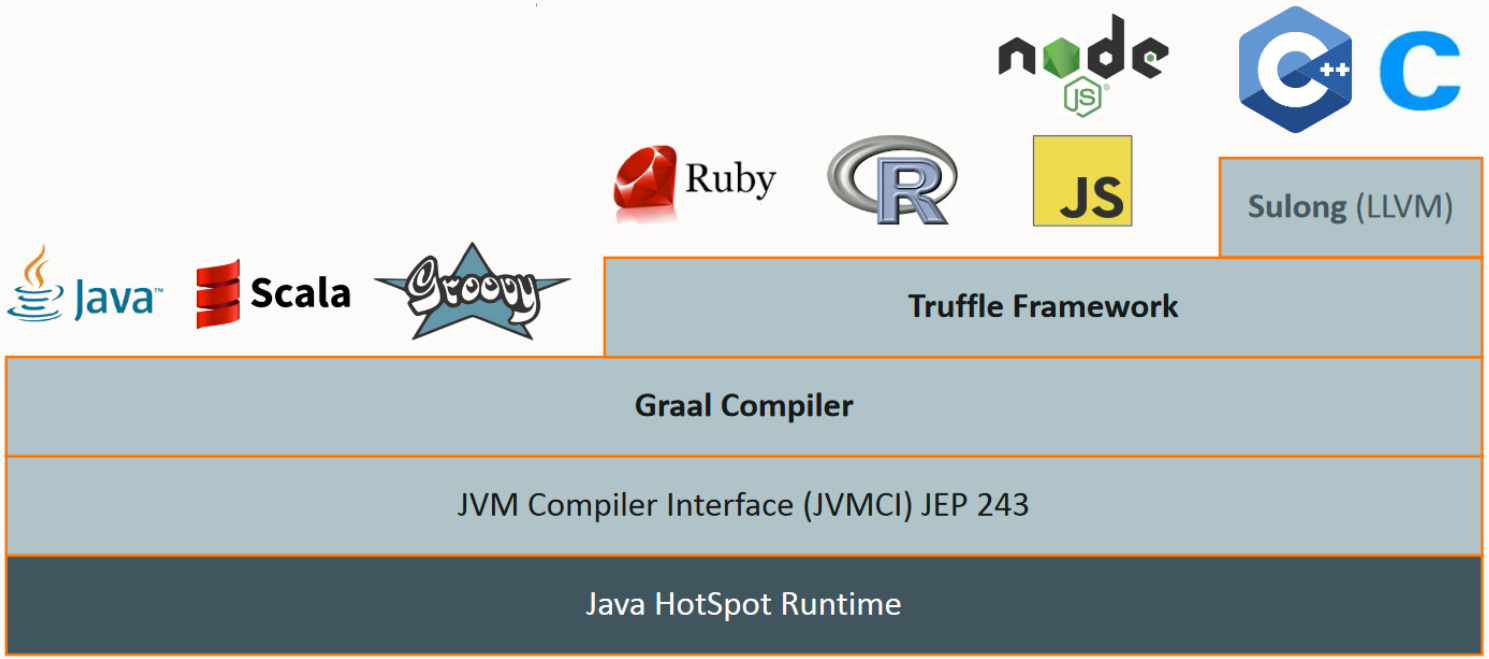
Da li kombinacija umetanja metoda i senzitivne analize daje još bolje performanse koda?



Copyright © 2021, Oracle and/or its affiliates | Confidential: Internal

Poliglotizam i GraalVM






```
app.get('/', function (req, res) {
  var text = 'Hello World from Graal.js!<br> '

  // Using Java standard library classes
  text += BigInteger.valueOf(10).pow(100)
          .add(BigInteger.valueOf(43)).toString() + '<br>'

  // Using R methods to return arrays
  text += Polyglot.eval('R',
      'ifelse(1 > 2, "no", paste(1:42, c="|"))') + '<br>'

  // Using R interoperability to create graphs
  text += Polyglot.eval('R',
    `svg();
     require(lattice);
     x <- 1:100
     y <- sin(x/10)
     z <- cos(x^1.3/(runif(1)*5+10))
     print(cloud(x~y*z, main="cloud plot"))
     grDevices:::svg.off()
    `);

  res.send(text)
})
```


Copyright © 2021, Oracle and/or its affiliates | Confidential: Internal

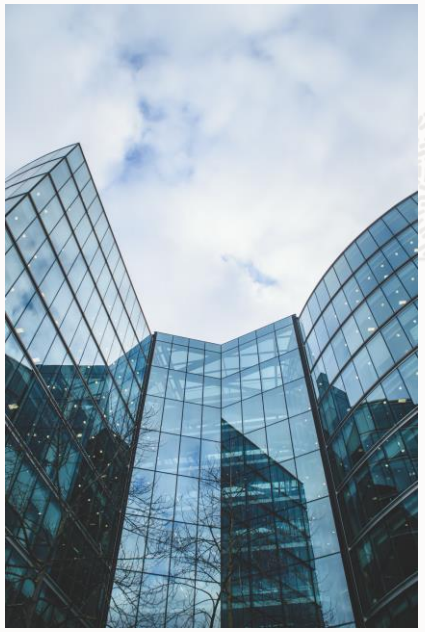
Espresso – Java on Truffle



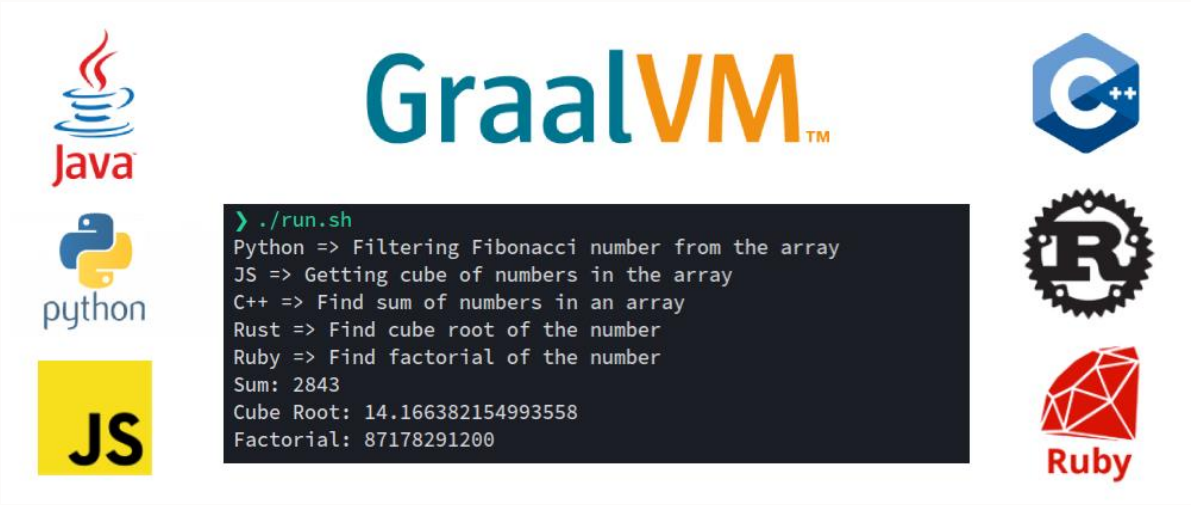
- Interpreter Java bajtkoda pisan u Javi, postao JIT kompilator koji koristi Truffle i Graal
- Moguće je koristiti Espresso za rad sa Javom u poliglot kontekstu ili kao bajtkod interpreter
- Espresso je i dalje prototip:
  - Slabije performance u odnosu na HotSpot VM
  - Ne podržava AOT kompilaciju
- Espresso AOT projekat:
  - Implementirati Truffle AOT mogućnosti za Espresso
  - Cilj: Kontekstno-nezavistan kod (već sad ~11% poboljšanja performansi)
- Budućnost?

Copyright © 2021, Oracle and/or its affiliates | Confidential: Internal

Priključivanje projektu i otvoreni problemi



- Prvi koraci prilikom uključivanja na projekat
- Otvoreni problemi



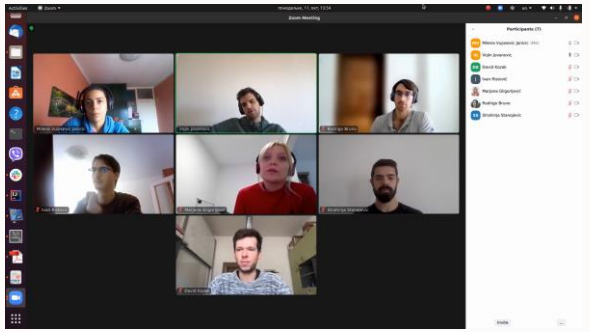


Copyright © 2021, Oracle and/or its affiliates | Confidential: Internal

Priključivanje projektu

- Upoznavanje sa projektom i instalacija potrebnih alata
- Sastanci na nedeljnom nivou (research, community, 1 na 1)
- Prvi zadatak
  - Serijalizacija lambda funkcija
- Code review
- Dalji koraci





Copyright © 2021, Oracle and/or its affiliates | Confidential: Internal

Otvoreni problemi: spekulativna serializacija i deserializacija podataka

U point-to-point komunikaciji struktuiranih podataka informacije se ponavljaju

- Stringovi imaju mali podskup svih mogucih vrednosti, npr. vlastita imena
- Brojevi su malom interval-u, npr. ocene studenata

Primaoc struktuiranih podataka pristupa poljima po redosledu

Problem: napraviti serializator i deserializator nezavistan od jezika koji:

- U prvoj fazi radi kao interpreter i skuplja informacije o obliku podataka
- Pored tipova podataka, skuplja i skup vrednosti: opseg celih brojeva, stringove koji se ponavljaju
- Deserializator meri frekvencije i redosled pristupanja poljima
- Serializator i deserializator u isto vreme generišu kod za komprimovanu serializaciju i deserializaciju
- Podaci se šalju u po redosledu pristupanja i mogu se čitati pre prispeća cele poruke
- Za kompresiju, nema lookup tabela već su tabele u samom genrisanom assembly kodu

Copyright © 2021, Oracle and/or its affiliates | Confidential: Internal

Otvoreni problemi: unifikacija sampling-a i instrumentacije

Skupljanje profila je neophodno za najbolje performanse

- Sampling uvodi nepreciznost uz minimalno usporenje (1%-3%)
- Instrumentacija je precizna ali unosi 10%-15% usporenje

Za kratke procese instrumentacija je bolja, dok je za dugacke procese sampling bolji

Profili se skupljaju za brojne tačke u kodu: kondicionale, virtualne pozive, monitore, pozive, itd.

Problem: definisati koji se profili skupljaju deklarativno i iz toga genrisati ili sampling ili instrumentaciju

Copyright © 2021, Oracle and/or its affiliates | Confidential: Internal

Zaključci i diskusija

- GraalVM je platforma otvorenog koda koja omogućava mnošto interesantnih istraživanja
- Predstavljeni su problemi na kojima se trenutno radi ali i primeri otvorenih tekućih problema
- U saradnji sa Matematičkim fakultetom, rad na projektu finansira Oracle Labs
- Ukoliko je neko zainteresovan za priključivanje projektu može da se javi
- Postoje otvorene pozicije za internship ali i za posao u okviru Oracle Labs-a, detalje potražite na http://oglasi.matf.bg.ac.rs/
- Novi rezultati uskoro

Pitanja?



Copyright © 2021, Oracle and/or its affiliates | Confidential: Internal