Projekat GraalVM

Seminarski rad u okviru kursa
Metodologija stručnog i naučnog rada
Matematički fakultet

Bojan Bardžić, Milica Gnjatović, Pavle Savić, Andrija Urošević
kontakt email prvog, drugog, trećeg, mi18083@alas.matf.bg.ac.rs

20. novembar 2022.

Sažetak

U ovom tekstu je ukratko prikazana osnovna forma seminarskog rada. Obratite pažnju da je pored ove .pdf datoteke, u prilogu i odgovarajuća .tex datoteka, kao i .bib datoteka korišćena za generisanje literature. Na prvoj strani seminarskog rada su naslov, apstrakt i sadržaj, i to sve mora da stane na prvu stranu! Kako bi Vaš seminarski zadovoljio standarde i očekivanja, koristite uputstva i materijale sa predavanja na temu pisanja seminarskih radova. Ovo je samo šablon koji se odnosi na fizički izgled seminarskog rada (šablon koji *morate* da koristite!) kao i par tehničkih pomoćnih uputstava. Pročitajte tekst pažljivo jer on sadrži i važne informacije vezane za zahteve obima i karakteristika seminarskog rada.

Sadržaj

Slika 1: Podržani jezici

1 Uvod

GraalVM je alat koji omogućava pisanje i izvršavanje koda u različitim jezicima. GraalVM podržava naredne jezike:

- Java
- JavaScript i Node.js
- Python
- Ruby
- R
- LLVM jezici poput C-a i C++-a
- WebAssembly

GraalVM je implementiran u javi.

Ovaj alat je koristan u radu sa mikroservisnom arhitekturom. Nekolicina okvira za rad sa Java mikroservisima je već prihvatila ovu platformu. Između ostalih to su Micronaut, Spring, Helidon i Quarkus.

Još jedna od mogućnosti koje GraalVM nudi je implemenitranje novih jezika i alata korišćenjem biblioteke Truffle.

Korišćenjem ovog alata je omogućeno efikasnije korišćenje više jezika na jednom projektu. U projektu je moguće kodom jednog jezika pozivati funkcije pisane u drugom jeziku. Dopušteno je deljenje struktura podataka između kodova pisanih u različitim jezicima. Na ovaj način je omogućeno da sakupljač otpadaka radi na celom projektu bez obzira na to koliko različitih jezika je korišćeno. Ovim je omogućeno i jednostavnije debagovanje.

GraalVM čine Java Virtuelna Mašina - JVM i Java Development Kit - JDK. Ovaj alat je proizvod kompanije Oracle. Osnovni cilj projekta je omogućiti brže izvršavanje koda, a da se pri tome koristi manje memorije.

Ovaj alat je dostupan na Linux, Windows i macOS operativnim sistemima.

Dostupna su dva izdanja GraalVM-a:

Community eddition je otvorenog koda i dostupno je u github repozitorijumu. Korisnici mogu doprineti razvoju ove verzije kreiranjem githb issue-a i pravljenjem pull request-ova.

Enterprise eddition razvija i licensira kompanija Oracle.

GraalVM podržavaju razvojna okruženja i protokoli za debagovanje. Neka od podržanih okruženja su Eclipse, NetBeans, IntellliJ IDEA i Visual Studio Code. Ova okruženja su posebno dobra jer podržaju sve jezike koje podržava i GraalVM. Ovaj alat obezbeđuje ugrađen Crome DevTools Protocol, Debug Adapter Protocol (DAP) i Language Server Protocol (LSP), čime je omogućeno debagovanje JavaScript, R i Ruby kodova.

1.1 Istorijski razvoj

Java Virtuelne mašine poput Oraklovog Java HotSpotVM i IBMov Java VM postoje već 20ak godina. Međutim nije postojala virtuelna mašina koja bi omogućila efikasno izvršavanje kodova pisani u raličitim jezicima. Cilj ovog projekta je bio da se napavi objedinjena virtuelna mašina koja bi ovo omogućila.

GraalVM 19.0 je objavljen u Maju 2019e i to je prva verzija ovog alata. Trenutno najnovija verzija je GraaLVM 22.1.0 objavljena u Aprilu 2022.

2 Kompanije koje koriste GraalVM

Facebook
: Društvena mreža Facebook ima ogroman broj korisnika zbog čega je jako bitno da se kod dobro skalira i brzo izvršava. Na serverskoj strani koristi Javu i Spark framework za rad sa velikom količnimo podataka. Prelazak sa Oracle JDK-a i Open JDK-a na GraalVM se sveo samo na promenu runtime okruženja, bez ikakvih promena u kodu. Izmereno je prosečno ubrzanje 1.1x korišćenjem Community verzije i ubrzanje 1.42x korišćenjem Enterprise verzije GraalVM-a.

Twitter
: Kao i Facebook, Twiter je društvena mreža sa milionima korisnika. Sa porastom broja korisnika bile su neophodne promene koje bi omogućile brzo izvršavanje uz minimalne troškove. Većina mikroservisa tvitera je implementirano u Skali. Prelaskom na GraalVM je omogućeno 8-11% ušteda na procesoru.

Standard Chartered Bank
: Ova kompanija je između ostalog koristila Javu, Python sa Spring-Boot okvirom kako bi obezbedila stabilnost i robusnost programa. Pre par godina kompanije je odlučila da svoju aplikaciju prebaci u cloud, što je donelo nove izazove. GraalVM je obezbedio jednostavan prelazak na cloud time što podržava tehnologije koje su programi već koristili. GraalVM je omogućio skalabilnost i ubrzanje od 7%.

Goldman Sachs
: Goldman Sachs je investiciona banka. Ova kompanija ima svoj programski jezik Slang koji ima funkcije pogodne za njiove potrebe. Jezik je dosta glomazan i potrebno mu je unapređenje. Prevođenje celog koda u neki drugi jezik bi bilo previše zahtevno. GraalVM i Truffle su omogućili unapređenje i ubrzanje izvršavanja koda pisanog u Slangu, a da je pri tom napisana minimalna količina novog koda.

Nvidia
: Nvdia je jedan od najvećih proizvođača grafičkih kartica. Iako neki jezici jednostavno mogu da ubrzaju izvršavanje korišćenjem grafičke

katice, za neke druge to može biti izazovno. GraalVM i Truffle su omogućili kreiranje grCuda jezika koji je kao dodatni jezik GraalVM. Kako jezici u GraalVM efikasno komuniciraju međusobno tako je omogućna jednostavna komunikacija sa jezikom grafičke kartice i samom grafičkom.

Politie
Holandska policija je imala monolitnu aplikaciju koja je koristila između ostalog TypeScrip, Angular, Scala, Axon, SpringBoot, Slick i R. Ova aplikacija je obrađivala velike količine podataka u realnom vremenu, ali to je bilo veoma sporo. Cilj je bio preći u cloud i na mikroservisnu ahitekturu, a pri tom ne implemenitrati sve iz početka. GraalVM je omogućio ovaj prelazak i ubrzanje, pri čemu je ostao značajan deo starog koda.

3 Šta obuhvata projekat?

Projekat GraalVM je istraživački projekat koji razvija Oracle labs. Od 2012. preko šezdeset naučnih radova je izdato od strane razvojnog tima. Sam projekat obuhvata više komponenti koje ga čine korisnim i inovativnim.

3.1 GraalVM kao zamena za JVM

Jedna od prednosti GraalVM je što se može koristiti umesto Java virtuelne mašine, on može da pokreće Javu, Skalu, Kotlin i sve ostale jezike koje se prevode u Java bajt kod. Od 2019. može se izvršavati na Linux-u, a od verzije 20.1.0 ima podršku i za Windows.

Još jedna od prednosti koja se pripisuje GraalVM-u su odlične performanse, neke demonstracije pokazuju da Ruby program izvršava i do 30 puta brže od originalne implementacije. Detaljnija testiranja su ipak pokazala da se u proseku Ruby kod izvšava oko 30% brže na GraalVM, što je i dalje obećavajući rezultat.

Kada su testirani Java programi rezultati su bili manje optimistični, performanse GraalVM-a su okvirno slične Oracle-ovom HotSpot kompajleru. Ovo samo po sebi ne predstavlja loš rezultat, ali ne predstavlja ni neki revolucionarni napredak u odnosu na sadašnje tehnologije.U svakom slučaju, GraalVM će biti brži od klaične JVM. Iako ne predstavlja poboljšanje u odnosu na HotSpot VM, prednost GraalVM-a je u tome što je nov kompajler nad kojim nije vršena optimizacija preko dvadeset godina, kao što je to slučaj sa HotSpot-om. Još jedna prednost u odnosu na prethodne kompajlere je to što je pisan u Javi za razliku od prethodnih kompajlera koji su pisani u jezicima C i C++, to omogućava lakše proširenje i optimizaciju od strane Java programera.

3.2 Truffle - kompajler za kompajlere

Truffle je biblioteka otvorenog koda za pravljenje alata i implementacija programskih jezika kao interpretera za samomodifikujuća apstraktna sintaksna stabla. On nam dozvoljava da pravimo interpretere bez većih problema, ali prednost njegovog korišćenja je u tome što se interpretirani kod izvršava podjednako brzo kao i kompajlirani kod. On je korišćen za

pisanje interpretera za jezik JavaScrpit u GraalVM-u. Pri izvršavanju koda potrebno mu je neko vreme da se zagreje, ali kada dostigne optimalne performanse one su okvirno iste kao V8 kompajler kompanije Google.

Sama optimizacija koda je zasnovana ideji parcijalne evaluacije. Truffle uzima interpreter koji smo mu dali i naš program i koristi JIT kompajler i promene sintaksnog stabla da izvrši optimizacije u našem kodu. Da bi se utvrdilo koje optimizacije je potrebno izvršiti program mora prvo biti pokrenut i zbog toga se javlja sporije vreme izvršavanja na početku.

Takođe, pri pisanju interpretera koristeći Truffle možemo da naglasimo gde i kada želimo da kod bude optimizovan. U nekim slučajevima promena u izvornom kodu može da dovede do greške u programu i u tom slučaju je potrebno deoptimizovati kod. Sve ove optimizacije i izbacivanja nekorišćenog koda iz našeg programa dovode do velike brzine izvršavanja.

Standardni podržani jezici za Truffle su JavaScript sa okruženjem Node.js ugrađenim u njega, kao i Ruby, R i Python. Pored toga, Truffle ima u sebe ugrađen Sulong koji može da izvršava LLVM bitkod i tako izvšava programe pisane na jezicima kao što su C, C++, C#, D, Lua i FORTRAN. Ovim omogućavamo da GraalVM izvšrava veliki broj različitih programskih jezika kao i da se funkcije iz jednog jezika pozivaju u drugom jeziku omogućavajući veću fleksibilnost pri pisanju programa.

3.3 Izvršavanje mašinskog koda na kaludu

Jedna od glavnih funkcionalnosti koje uvodi ovaj projekat jeste kompajliranje Java programa sa bajt koda na mašinski kod. Ovo je novina koja nam dozvoljava tako nešto prvi put od kako postoji programski jezik Java. Jedan od slučajeva upotrebe za ovo je izvšavanje mikroservisa na klaudu.

Mikroservisi se sada kompajliraju u izvršni fajl i time je eliminisana potrebna da na serverskoj strani mora postojati Java viruelna mašina, što dovodi do smanjenja zauzeća memorije na serveru. Još jedna prednost ovog pristupa je u tome što više nije potrebno da se za naše programe učitava mnoštvo Java klasa pri pokretanju, od kojih većina neće biti upotrebljeno. Ovo dovodi do bržeg izvršavanja naših programa na serverskoj strani i sprečava sporo izvšavanje pri prvom pokretanju.

4 Karakteristike

4.1 Native Image

4.2 Napredni optimizator

4.3 Polygot

4.4 Napredni alati

5 Zaključak

Literatura

A Dodatak

Ovde pišem dodatne stvari, ukoliko za time ima potrebe. Ovde pišem dodatne stvari, ukoliko za time ima potrebe. Ovde pišem dodatne stvari, ukoliko za time ima potrebe. Ovde pišem dodatne stvari, ukoliko za time ima potrebe. Ovde pišem dodatne stvari, ukoliko za time ima potrebe.