WETENSCHAPPEN & BIO-INGENIEURSWETENSCHAPPEN

De Onderzoeksgroep

Software Languages Lab

nodigt U graag uit op de openbare verdediging van het proefschrift van

Matteo Marra

ter behaling van de graad van Doctor in de Wetenschappen

Titel van het proefschrift:

A live debugging approach for Big Data processing applications

Promotor:
Prof. dr. Elisa Gonzalez Boix (VUB)

Co-promotor:
Dr. Guillermo Polito (Université de Lille)

De verdediging heeft plaats op

Dinsdag 3 mei 2022 om 17u in aula I.2.03

Please contact Matteo Marra at mmarra@vub.be if you want to join the presentation through Microsoft Teams

Samenstelling van de jury

Prof. dr. Geraint Wiggins (VUB, voorzitter)
Prof. dr. Bas Ketsman (VUB, secretaris)
Prof. dr. Jan De Beule (VUB)
Prof. dr. Guido Salvaneschi (University of St. Gallen)
Prof. dr. Luc Fabresse (IMT Nord Europe)
Prof. dr. Miryung Kim (University of California)

Curriculum vitae

Matteo Marra heeft in 2015 zijn Bachelor-diploma behaald aan de Universiteit van Bologna en in 2017 zijn Master-diploma aan de Vrije Universiteit Brussel (VUB). Hierna startte hij een doctoraat aan het Software Languages lab (SOFT) van de VUB met een beurs van het FWO in samenwerking met het RMoD-onderzoekslabo van INRIA Lille Nord France.

Zijn onderzoek richtte zich voornamelijk op geavanceerde debugging technieken voor Big Data processing applicaties om ontwikkelaars te helpen fouten op te lossen in uiterst parallelle en op afstand uitgevoerde applicaties. Matteo's onderzoek resulteerde in vier publicaties in internationale peer-reviewed tijdschriften en conferenties, en vier bijdragen aan internationale peer-reviewed workshops.

Abstract van het doctoraatsonderzoek

De moderne wereld is sterk afhankelijk van data: in 2020 zal wereldwijd meer dan 64 ZBs aan data worden gecreëerd, vastgelegd, gekopieerd of verbruikt. Als gevolg daarvan zijn nieuwe softwareplatforms ontstaan om grote datasets uit verschillende domeinen op een parallelle en schaalbare manier te analyseren. De twee meest prominente programmeringsmodellen voor Big Data-verwerking zijn Map/Reduce en Apache Spark. Beide modellen richten zich op het programmeren van complexe problemen via vertrouwde functies, en laten het framework de distributieaspecten afhandelen, bijvoorbeeld de parallellisatie en de fouttolerantie voor node-fouten.

Het debuggen van Big Data-toepassingen is moeilijk door hun gedistribueer\-de en parallelle aard, waardoor de afstand tussen de hoofdoorzaak van een bug en een waargenomen fout groter is. Bovendien hebben ontwikkelaars de neiging om een grote technologiestack te gebruiken, wat het debuggen ook bemoeilijkt. Een gebruikelijke debugging methode is het analyseren van logfiles, maar deze missen contextuele informatie over welke record(s) een fout veroorzaakte(n). Recentelijk zijn Record \& Replay debuggers bestudeerd, maar het opnieuw afspelen van Big Data applicaties kan erg kostbaar zijn omdat ze normaal gesproken lang duren. Checkpoint-gebaseerde debugging technieken kunnen in principe de replay tijd verminderen, maar vereisen steeds de creatie van een checkpoint en een replay stap.

In deze dissertatie onderzoeken we geavanceerde online debugging oplossingen afgestemd op Map/Reduce en Spark-achtige programma's. Eerst stellen we out-of-place debugging voor, een nieuwe debugging architectuur om remote en gedistribueerde applicaties te debuggen. In dit model, wanneer er een fout optreedt in een applicatie die op afstand draait (bv. in een cluster), wordt de staat van de berekening overgebracht naar de machine van de ontwikkelaar, m.a.w. naar de locatie waar de applicatie gedebugged kan worden. Dit vermijdt het opnieuw herhalen van de uitvoering, terwijl het een volledig interactieve debug-omgeving biedt.

We verkennen vervolgens de toepasbaarheid van out-of-place debugging op parallelle en gedistribueerde Map/Reduce en Spark-achtige applicaties, door middel van twee nieuwe technieken voor het optimaliseren van het debug process: composite debugging events, dat wil zeggen het groeperen en gecentraliseerd debuggen van meerdere soortgelijke debugging events, en dynamische lokale checkpoints, dat wil zeggen het dynamisch vastleggen van de uitvoeringsstatus. Zo kunnen we Big Data-applicaties gecentraliseerd vanop afstand debuggen, en breiden we het uit met domeinspecifieke debugging-operaties. Ten slotte complementeren we onze debugging-aanpak met een versoepeld computationeel model waarmee ontwikkelaars de runtime kunnen opdragen om automatisch een gedefinieerde hoeveelheid uitzonderingen te negeren die tijdens runtime optreden. Deze functie is vooral relevant voor data-analyse toepassingen die een verlies in nauwkeurigheid kunnen accepteren (bijv. als gevolg van vervuilde gegevens).

We implementeren onze debugging technieken in Pharo Smalltalk, bovenop Port en Spa, onze frameworks die respectievelijk het Map/Reduce en het Spark-achtige model implementeren. Verder hebben we alle call-stack operaties die nodig zijn om onze debugging aanpak mogelijk te maken veralgemeend in Sarto, een call-stack instrumentatie laag voor stack tailoring. De voorgestelde out-of-place debugging aanpak toegepast op het debuggen van Map/Reduce en Spark-achtige programma's, samen met Sarto, vertegenwoordigen de belangrijkste bijdragen van dit proefschrift.

Onze validatie is tweevoudig: we valideren onze debugging aanpak kwantitatief en kwalitatief. Voor de kwantitatieve studie hebben we performantie benchmarks uitgevoerd die aantonen dat ons model schaalt naar een toenemende hoeveelheid van zowel data als parallelle uitzonderingen. Voor de kwalitatieve studie hebben we een gebruikersonderzoek uitgevoerd om de bruikbaarheid van onze aanpak voor het oplossen van verschillende debugging taken te beoordelen en deze te vergelijken met een reproductie van een state-of-the-art debugger voor Spark applicaties. De resultaten tonen aan dat de deelnemers een betere debugging-ervaring rapporteerden door gebruik te maken van onze debugger, en dat ze de geavanceerde functies die onze debugger biedt positief waardeerden.