Publicatie : AG Connect
Datum : 31 dec 2020 Regio : Nederland Pagina : 67 Bereik : -1
cm2 : 1350 Frequentie : 10x per jaar AVE : € 17150,00 Oplage : 8.000

IT-STUDIES

Typesystemen toegespitst op data én interacties als oplossing

PROGRAMMEREN VOOR MULTICORE KAN MAKKELIJKER

HET ONTWIKKELEN VAN MULTICORE SOFTWARE IS NIET POPULAIR ONDER PROGRAMMEURS. VOORAL HET PROGRAMMEREN VAN INTERACTIES TUSSEN REKENTAKEN IS COMPLEX. EEN BELANGRIJKE OORZAAK IS DAT VEEL ONTWIKKELAARS NOG MET OUDE PROGRAMMEERTECHNIEKEN WERKEN DIE MINDER GESCHIKT ZIJN VOOR DE ONTWIKKELING VAN MULTICORE SOFTWARE. VOLGENS SUNG-SHIK JONGMANS KUNNEN TYPESYSTEMEN DIE GEHEEL ZIJN TOEGESPITST OP DATA ÉN INTERACTIES HIERVOOR EEN GOEDE OPLOSSING BIEDEN.

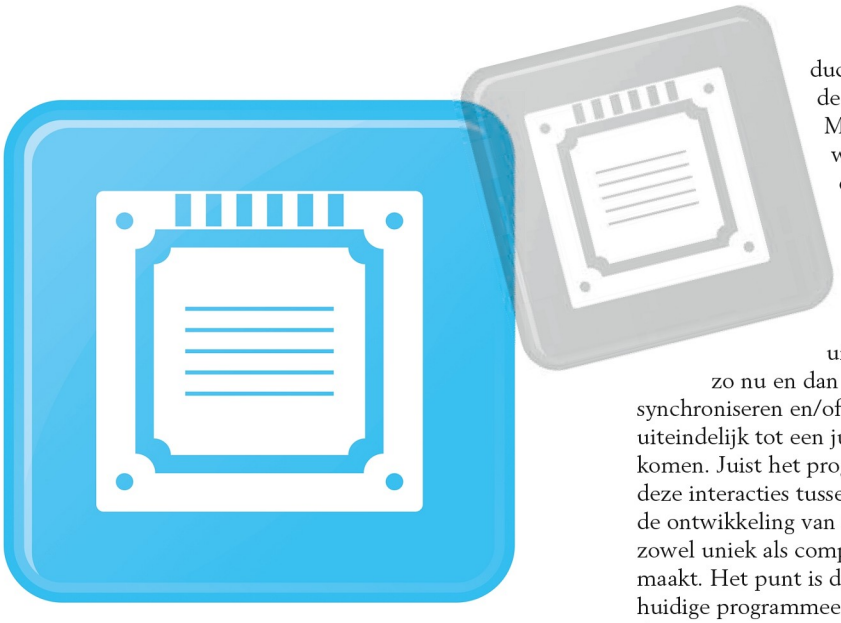
door Sung-Shik Jongmans beeld Shutterstock

WIE VANDAAG VOOR HET EERST IN TWEE DECENNIA EEN NIEUWE COMPUTER KOOPT, KAN ZICH VERBAZEN: PROCESSORS draaien nog steeds op een vergelijkbare kloksnelheid als aan het begin van deze eeuw. Wat is er in de tussentijd gebeurd met de exponentiële snelheidsgroei die we sinds de jaren zeventig gewend waren? Hebben processorfabrikanten de afgelopen twintig jaar stilgezeten? Nee, bepaald niet. In tegenstelling tot processors uit de vorige eeuw hebben moderne processors niet één core maar meerdere. De kloksnelheden zijn weliswaar vergelijkbaar, maar per seconde kunnen N cores natuurlijk N keer zoveel werk gedaan krijgen. Bedrijven als Intel en AMD produceren inmiddels processors met meer dan 25 cores, ook voor de 'gewone' consument. En dat aantal zal alleen maar toenemen. Over een paar jaar worden er dus machines verkocht die in potentie honderd tot duizend keer krachtiger zijn dan de pc's van twintig jaar geleden. In theorie, althans.

PROGRAMMEERTECHNIEKEN VAAK NOG UIT DE VORIGE EEUW

De praktijk is weerbarstiger. In het algemeen is krachtige hardware pas echt waardevol in combinatie met software die hem optimaal kan benutten. En daar wringt de schoen. Het gerenommeerde onderzoeksbureau Gartner beschreef de

Publicatie : AG Connect
Datum : 31 dec 2020 Regio : Nederland Pagina : 67 Bereik : -1
cm2 : 1350 Frequentie : 10x per jaar AVE : € 17150,00 Oplage : 8.000



AUTEUR

SUNG-SHIK JONGMANS
is universitair docent/onderzoeker bij de Open Universiteit en het CWI.

situatie al in 2017, enigszins alarmerend: "Multicore programming is generally seen as a hard-to-achieve and time-consuming task, so many programmers avoid it as far as possible."
Structurele onderbenutting van hardware – cores die niets doen – ligt dus op de loer. Tegelijkertijd heeft het voor de consument weinig meerwaarde om hardware met tientallen cores te kopen als de software er slechts enkele van kan gebruiken.

HET PROBLEEM MET MULTICORE SOFTWARE

Wat is er zo moeilijk aan de ontwikkeling van multicore software? Hier is een analogie. Stel dat we een team (= processor) hebben met meerdere leden (= cores) die een bepaald project (= software) moeten uitvoeren. Elk van de leden heeft zijn eigen taken en klusjes. Daarnaast bestaan tussen de werkzaamheden van verschillende leden ook allerlei afhankelijkheden. Bijvoorbeeld: als Alice een deelproduct oplevert en Bob verantwoordelijk is voor de kwaliteitscontrole, dan zal Alice het deelproduct nog niet mogen doorsturen naar Carol voordat Bob de controle heeft uitgevoerd. Anders voldoet het eindproduct mogelijk niet aan de eisen.
Multicore software werkt op precies dezelfde manier: er is een aantal rekentaken, dat verspreid over de cores van de processor wordt uitgevoerd, maar dat zo nu en dan met elkaar moet synchroniseren en/of communiceren om uiteindelijk tot een juiste uitkomst te komen. Juist het programmeren van deze interacties tussen rekentaken is wat de ontwikkeling van multicore software zowel uniek als complex/foutgevoelig maakt. Het punt is dat veel van de huidige programmeertechnieken nog uit de vorige eeuw stammen. Het is daarom niet zo verwonderlijk dat deze technieken minder geschikt blijken te zijn voor de ontwikkeling van moderne multicore software.

TYPESYSTEMEN VOOR INTERACTIEPATRONEN

Veel populaire programmeertalen, zoals Java, C#, en TypeScript, maken gebruik van types om het programmeerwerk te versimpelen. Het idee is dat de programmeur voor elk stuk data in het programma expliciet aangeeft wat voor soort informatie het betreft. Bijvoorbeeld: het stuk data is een getal, of een tekstfragment, of een afbeelding. Vervolgens kan een tool automatisch de code analyseren om programmeerfouten te ontdekken. Bijvoorbeeld: een getal mag nooit worden opgeteld bij een tekstfragment. Zónder types wordt de tester (of eindgebruiker!) pas met zulke fouten geconfronteerd wanneer het programma wordt uitgevoerd; mét types wordt de programmeur al veel eerder over deze fouten geïnformeerd, nog tijdens het schrijven van de code.
En dat is heel fijn: hoe eerder fouten worden ontdekt, hoe eenvoudiger en goedkoper het is om ze te herstellen. Typesystemen voor data bestaan al sinds

Publicatie : AG Connect
Datum : 31 dec 2020 Regio : Nederland Pagina : 67 Bereik : -1
cm2 : 1350 Frequentie : 10x per jaar AVE : € 17150,00 Oplage : 8.000

de jaren vijftig en worden in de praktijk dagelijks door tienduizenden programmeurs gebruikt. Gezien dit succes is de volgende onderzoeksvraag dan ook een logische: kunnen we eenzelfde soort techniek ontwikkelen voor multicore software, en dan volledig toegespitst op data én interacties?
Wereldwijd zijn diverse onderzoeksgroepen op zoek naar antwoorden op deze vraag. Het centrale idee is dat de programmeur niet alleen types voor data opgeeft, maar ook voor interactiepatronen. Dat wil zeggen dat de programmeur wordt gevraagd om in de code aan te geven welke soorten data op welke momenten tussen welke rekentaken mogen worden uitgewisseld. Vervolgens kan de programmeur, vergelijkbaar met klassieke datatypes (maar dan een stuk complexer!), tools gebruiken om automatisch te controleren of de rekentaken de interactiepatronen volgen. Is dat niet het geval, dan is er iets mis met het programma en wordt de programmeur hier direct over geïnformeerd.
Het mooie is dat deze techniek weinig extra moeite kost voor de programmeur, terwijl een belangrijke klasse van fouten snel, effectief en automatisch kan worden gedetecteerd en gediagnosticeerd. Bovendien is de codeanalyse volledig losgekoppeld van de executie; hierdoor wordt executieoverhead vermeden en is het ook nog eens mogelijk om aanvullende optimalisaties toe te passen.
Er zijn momenteel al diverse prototypes van deze techniek ontwikkeld (inclusief tools) voor een aantal populaire programmeertalen.

NWO RUBICON EN IMPERIAL COLLEGE LONDON

Sung-Shik Jongmans is sinds 2010 werkzaam als informaticawetenschapper: eerst als promovendus bij de Universiteit Leiden en het Centrum Wiskunde & Informatica (CWI), nu als universitair docent/onderzoeker bij de Open Universiteit en het CWI. Zijn onderzoeksdoel is om het ontwikkelen van multicore software eenvoudiger te maken. Sinds een aantal jaar is hij vooral gefascineerd door geavanceerde typesystemen.
Zijn onderzoek is deels gefinancierd door een subsidie in het Rubicon-programma van NWO (Nederlandse Organisatie voor Wetenschappelijk Onderzoek). Rubicon-subsidies zijn persoonlijke beurzen om jonge wetenschappers onderzoekservaring op te laten doen bij topinstellingen in het buitenland. Jongmans is met deze beurs twee jaar (2017-2019) gedetacheerd geweest bij Imperial College London, in een zeer sterke onderzoeksgroep met veel speciale expertise op het gebied van typesystemen.
Hij richtte zich daar vooral op de ontwikkeling van typesystemen voor interactiepatronen met geavanceerde control-flow; dit is een erg belangrijk aspect om dit soort technieken in de praktijk breed te kunnen inzetten. Op basis van een nieuwe theorie heeft hij, samen met collega's, een prototype gemaakt en geëvalueerd. Hieruit bleek dat hun techniek inderdaad krachtig genoeg is voor meerdere realistische casestudies. Deze bemoedigende onderzoeksresultaten zijn inmiddels gepubliceerd en gepresenteerd op prestigieuze internationale wetenschappelijke conferenties.
Jongmans: "Inhoudelijk was het voor mij heel interessant en leerzaam om dagelijks met topexperts te mogen samenwerken, overleggen en discussiëren. Het netwerk dat ik op deze manier heb kunnen opbouwen en de kennis die ik heb opgedaan, zijn tot de dag van vandaag van grote waarde voor mij. Sterker nog: zonder die expertise had ik mijn huidige onderzoeksproject – ook gefinancierd door NWO, in het Veni-programma – niet kunnen bedenken, opzetten en uitvoeren. Deze kennis was nog niet beschikbaar in Nederland toen ik mijn Rubicon-aanvraag bij NWO indiende; dit was een belangrijke motivatie voor mij om voor Imperial College London te kiezen."
"Mijn verblijf in Londen was niet alleen inhoudelijk een verrijking, maar ook wat academische cultuur betreft. Zo ervaarde ik bijvoorbeeld dat de wetenschappelijke mentaliteit bij Imperial College London wat meer hiërarchisch en competitief is dan wat ik bij onderzoeksinstellingen in Nederland heb meegemaakt. Het was boeiend om daar een tijdje in mee te mogen draaien!"