Intreerede
prof.dr.ir. Dick Epema
23 november 2012

/ Faculteit Wiskunde & Informatica

TU/e Technische Universiteit **Eindhoven** University of Technology

Decentraliseer – en beheers?

Where innovation starts

Prelude: vrienden van vrienden

Al jarenlang houdt NWO, de Nederlandse Organisatie voor Wetenschappelijk Onderzoek, de zogeheten Wetenschapsquiz. De vragen van deze quiz worden in de Nederlandse dagbladen gepubliceerd en de antwoorden worden gegeven in een nogal ludiek tv-programma met de nodige, vaak experimentele, uitleg. Tot mijn genoegen bevatte deze quiz in 2011 een vraag die als een informaticavraag kan worden aangemerkt, en meer nog, die raakt aan de kern van het onderwerp van deze rede, te weten decentralisatie.

De vraag luidde als volgt:

> *"Je kunt bij Facebook heel goed zien hoeveel vrienden jouw vrienden hebben. Hebben mensen op Facebook gemiddeld net zoveel vrienden als hún vrienden?"*

De mogelijke antwoorden waren:

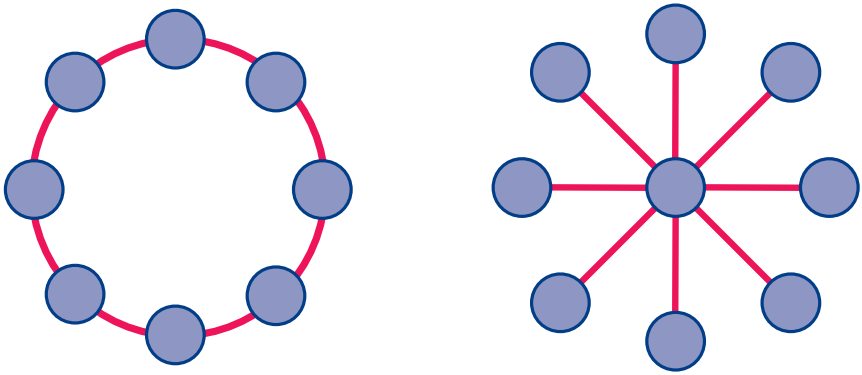
a. *Ja*
b. *Nee, gemiddeld hebben hun vrienden meer vrienden dan zij*
c. *Nee, gemiddeld hebben hun vrienden minder vrienden dan zij*

Laten we, om enige intuïtie te ontwikkelen voor het goede antwoord op deze vraag, kijken naar twee gevallen van netwerken van mensen met vriendschapsrelaties. In het eerste geval staat een groep mensen hand in hand in een kring en houdt iedereen de hand vast van zijn of haar twee vrienden. Dit netwerk is voor acht personen symbolisch weergegeven in Figuur 1 (links). Ieder van de twee vrienden van willekeurig welke persoon heeft uiteraard ook weer twee vrienden, dus het gemiddelde aantal vrienden van vrienden is gelijk aan twee. We kunnen dus concluderen dat in dit speciale geval het juiste antwoord op de vraag 'Ja' is. In het algemeen is het natuurlijk direct duidelijk dat als iedereen evenveel vrienden heeft, bij voorbeeld als iedereen vriend is van iedereen, het antwoord 'Ja' is. Overigens kan de ringstructuur waarin de personen opgesteld staan en waarin iedereen een soortgelijke rol speelt, een geheel *gedecentraliseerde* interconnectiestructuur genoemd worden.

Laten we nu een netwerk met een andere, sterk *gecentraliseerde*, interconnectiestructuur beschouwen. Stel dat er één centrale persoon is, een allemansvriend, die vriend is van alle andere personen, en dat alle andere personen slechts één vriend hebben, de allemansvriend. Dit netwerk is voor negen personen weergegeven in Figuur 1 (rechts). Laten we nu de twee benodigde grootheden, het gemiddelde aantal vrienden en het gemiddelde aantal vrienden van vrienden, in dit geval berekenen. Stel het totale aantal personen wordt weergegeven door het symbool K. De centrale persoon heeft $K–1$ vrienden en ieder van de $K–1$ andere personen heeft er slechts één, zodat het gemiddelde aantal vrienden gelijk is aan:

$$[1 \times (K–1) + (K–1) \times 1] / K = 2 – (2/K),$$

wat voor een groot netwerk (een grote waarde van K) neerkomt op ongeveer twee.



Figuur 1

Een gedecentraliseerde (links) en een gecentraliseerde (rechts) interconnectiestructuur.

Voor de vrienden van vrienden wordt het in dit geval allemaal iets moeilijker. De centrale persoon heeft $K–1$ vrienden die allemaal één vriend hebben, namelijk hem. Ieder van de $K–1$ anderen heeft slechts één vriend, maar die heeft wel $K–1$ vrienden. In totaal hebben alle personen samen $2K–2$ vrienden. Dit alles combinerend is het gemiddelde aantal vrienden van vrienden dus gelijk aan:

$$[1 \times (K–1) \times 1 + (K–1) \times 1 \times (K–1)] / (2K–2) = K/2,$$

ofwel de helft van het aantal personen in de groep. De conclusie in dit geval is dat het juiste antwoord dus antwoord b. is: *Nee, gemiddeld hebben hun vrienden meer vrienden dan zij*. In het algemeen is het zelfs zo dat zodra niet iedereen hetzelfde aantal vrienden heeft, het juiste antwoord b. is [10].

Het zal nu duidelijk zijn dat b. het juiste antwoord op de quizvraag is. In het dagelijks leven – en Facebook behoort tegenwoordig tot het dagelijkse leven! – zullen sommige mensen veel meer vrienden maken dan andere. Deze populaire figuren werken heel sterk door in het gemiddelde aantal vrienden van vrienden doordat ze van velen vriend zijn en doordat ze veel vrienden hebben. Het verschijnsel dat aan deze vraag ten grondslag ligt, wordt wel de 'vriendschapsparadox' genoemd [10]. Wat we uit deze quizvraag kunnen concluderen is dat differentiatie in lokale, persoonlijke eigenschappen tot centralisatie, of althans tot reductie van decentralisatie, leidt.

Helaas bevat de wetenschapsquiz van NWO vaak erg veel wiskundige en natuurwetenschappelijke vragen – altijd drijft, zinkt, bevriest of smelt er wel iets – maar vrijwel nooit informaticavragen. Dit jaar is de quiz alweer verschenen, maar ik roep alle informatici in Nederland op de komende jaren veel vragen bij NWO in te dienen. Informatica onder de aandacht van het grote publiek brengen, op een andere manier dan via veiligheidslekken in computers of uit de hand gelopen automatiseringsprojecten, zal het imago van het vakgebied goeddoen.

Overigens heeft de vraag hierboven een serieuze achtergrond. Ten eerste is een *Online Social Network* als Facebook erg geïnteresseerd in allerlei eigenschappen van het netwerk tussen haar leden, bijvoorbeeld om hen gericht advertenties voor te schotelen of nieuwe vriendschappen te suggereren. Facebook bezit voor de opslag en analyse van haar ledennetwerk dan ook grote *data centers*, fabrieks-hallen vol met computers. Ten tweede komen de analyse van netwerkstructuren en het efficiënt gebruik van *data centers* terug in de twee onderzoeksgebieden waar ik mij mee bezighoudt: *Peer-to-Peer*-systemen en *Online Social Networks* enerzijds en *scheduling* in grote computersystemen anderzijds. In het vervolg van deze rede geef ik een schets van deze twee onderzoeksgebieden en stip ik een paar specifieke problemen daarin aan, afgewisseld met intermezzo's over het begrip 'decentralisatie' en over experimenten in de informatica.

Online Social Networks en Peer-to-Peer-systemen

Ik zal beginnen met het schetsen van een beeld van het onderzoek en de ontwikkeling in het onderzoeksgebied *Online Social Networks* (OSNs) en *Peer-to-Peer*(P2P)-systemen. OSNs zoals Facebook – en er zijn er tegenwoordig heel veel van, hoewel geen enkele de omvang van Facebook (in september 2012 1 miljard leden!) ook maar benadert – zijn in principe bij uitstek gedecentraliseerde systemen: uit vrije wil worden personen lid en gaan zij vriendschappen aan. Zoals we hiervoor al hebben gezien, kan zo'n systeem worden voorgesteld als een netwerk met een knoop voor iedere persoon en een directe verbinding tussen twee knopen als de corresponderende personen vrienden zijn. Een interessant onderzoeksonderwerp in dit domein is welke netwerkstructuren er in de praktijk ontstaan. Eén van de vragen die over zo'n netwerk gesteld kunnen worden heb ik aan het begin van deze rede behandeld.

OSNs zijn dus naar hun aard gedecentraliseerde systemen. Maar wat betreft de computersystemen waarmee ze worden ondersteund, zijn ze dat vrijwel nooit, althans voor zover het de grotere en succesvolle betreft. Zoals net al vermeld, gebruikt Facebook gecentraliseerde *data centers*. Vanuit een fundamenteel informatica-oogpunt is dit erg oninteressant. Een decentrale implementatie, waarbij de voor een gebruiker relevante gegevens alleen op zijn eigen computer, of op enkele welgekozen computers van bijvoorbeeld een paar van zijn vrienden, opgeslagen zouden zijn, is veel uitdagender. Er bestaan wel enkele van dergelijke systemen, zoals Diaspora* [1], maar ze hebben weinig gebruikers of ze zijn alleen experimenteel. Een moeilijk probleem dat in dergelijke systemen moet worden opgelost, is dat van de beschikbaarheid: gegevens die nodig zijn of bijgewerkt moeten worden, maar opgeslagen zijn bij vrienden die tijdelijk *offline* zijn, zijn niet bereikbaar. Een belangrijke conclusie, die zeker niet alleen hier geldt, is dat computersystemen in de échte, commerciële wereld zich vaak conformeren aan het adagium 'centraliseer wat kan, en decentraliseer wat moet'. Informatica-onderzoekers streven juist liever het omgekeerde na. Gedecentraliseerde OSNs zijn dus voor hen een interessant onderzoeksonderwerp, ook vanuit privacy-overwegingen.

P2P-systemen zijn naar hun aard én naar hun implementatie gedecentraliseerd. Zij worden voornamelijk gebruikt voor het uitwisselen van muziek- en video-

bestanden, waarvoor de computers van gebruikers direct met elkaar communiceren. P2P-systemen ontstonden eind jaren 90 vooral doordat door de invoering van ADSL de internetverbindingen van de computers van thuisgebruikers veel sneller werden. Om hiervan een illustratie te geven, in 1988 installeerde ik thuis mijn eerste modem met een snelheid van 1200 bits/seconde, terwijl mijn huidige modem een snelheid van 8 Megabit/seconde heeft. Het ophalen van een bestand van 500 MegaByte, een normale omvang voor een speelfilm, zou in 1988 maar liefst 40 dagen hebben geduurd, nog afgezien van alle onderbrekingen van de verbinding die vermoedelijk zouden zijn opgetreden. Bovendien zou mijn telefoon gedurende die tijd niet bruikbaar zijn geweest omdat het een *dial-up* modem betrof. Met mijn huidige modem zou het oversturen van hetzelfde bestand (theoretisch) krap tien minuten duren. Als deze ontwikkeling zich onveranderd voortzet – maar ik ben geen netwerkexpert, dus ik durf het niet te voorspellen – dan duurt het in 2036, na nogmaals 24 jaar, één tiende seconde om dezelfde film binnen te halen.

Figuur 2
Modems uit 1988 en 2012.

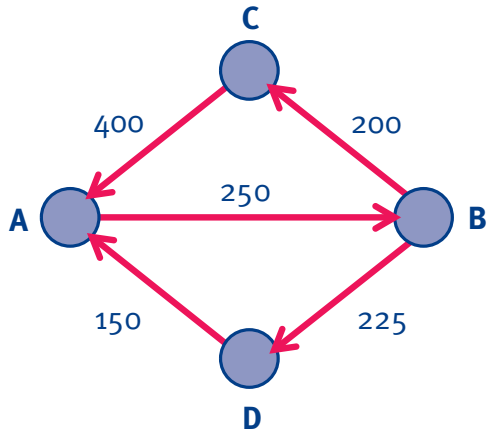
Het meest gebruikte P2P-systeem van het afgelopen decennium is BitTorrent, dat honderden miljoenen gebruikers heeft en circa 18% van de capaciteit van de internet *backbone* gebruikt. BitTorrent hakt bestanden in stukken en de *peers* wisselen deze stukken via een effectief *tit-for-tat* (voor wat hoort wat)-algoritme met elkaar uit. BitTorrent is een vruchtbare bron gebleken voor allerlei varianten en uitbreidingen, met daaraan gerelateerd prestatieanalyse-onderzoek, waarvan wij ook veel hebben gedaan (en nog doen) aan de TU Delft. Maar daar waar we nu ingenieuze P2P-technieken bedenken om bestanden uit te wisselen, kunnen we volgens de bovenstaande berekening over een aantal jaar simpelweg en rechtstreeks in een flits hetzelfde resultaat bereiken. Dit ondersteunt mijn adagium dat we in de basisprogrammering van computersystemen vooral bezig zijn om te proberen

de beperkingen van de huidige *hardware* te omzeilen of te verbergen. Veel van de daarbij ontwikkelde technieken, denk bijvoorbeeld ook aan virtueel geheugen, zijn na enige (tientallen) jaren vaak niet meer relevant voor de werking van eigentijdse computers – maar nog wel interessant uit wetenschappelijk oogpunt!

In dezelfde tijd dat we aan de TU Delft in 2004 ons P2P-onderzoek begonnen, ging ook de gratis videodienst YouTube van start. Daarnaast bestaan er nu ook betaalde videodiensten, waarvan het Amerikaanse bedrijf Netflix op dit moment de belangrijkste exponent is. De door Netflix verstuurde video's gebruiken zo'n 25% van de capaciteit van de internet *backbone* in de VS. In juni van dit jaar verstuurde Netflix in de VS voor 1 miljard uur aan video's, wat neerkomt op gemiddeld 3 uur per hoofd van de bevolking. Van beide, Youtube en Netflix, is de gebruikte infrastructuur ook al erg gecentraliseerd! Het idee dat voor grootschalige distributie van video over het internet, inclusief tv-programma's, wel van P2P-technieken gebruikgemaakt zou moeten worden, is dus danig gelogenstraft.

Recent hebben we met een groepje promovendi de focus van ons P2P-onderzoek voor een deel verlegd naar vragen die met reputaties in *online* netwerken samenhangen [6], wat een notoir moeilijk probleem is. De reputatie van een *peer* in een *file sharing*-netwerk zal hoog zijn als de *peer* meer data naar andere *peers* stuurt dan hij ontvangt. Een *peer* zou reputaties kunnen gebruiken door alleen data te sturen naar een andere *peer* als diens reputatie hoog genoeg is. De vraag is dan hoe een *peer* de reputaties van de andere *peers* berekent. In onze huidige methode bekijkt hij daartoe het netwerk met de *peers* in het systeem als knopen en de voltooide data-overdrachten tussen *peers* als gerichte verbindingen. In dat netwerk berekent hij de maximale datastroom tussen hem en iedere andere *peer* in beide richtingen via alle mogelijke wegen in het netwerk. Indien de totale datastroom van een andere *peer* naar hem groter is dan andersom, kent hij een goede reputatie aan die *peer* toe. In Figuur 3 wordt een voorbeeld gegeven van deze manier van het berekenen van reputaties.

Het blijkt dat deze reputatieberekening verbeterd kan worden als een *peer* een niet erg centrale plaats in het netwerk inneemt, ofwel als hij een lage tussencentraliteit heeft; hier hebben we een analyse van netwerkstructuren nodig. De *tussencentraliteit* (Engels: *betweenness centrality*) van een knoop is gedefinieerd als de som over alle paren knopen in het hele netwerk van de fractie van kortste wegen die door de betreffende knoop gaan. In het eenvoudige geval dat er slechts één kortste weg bestaat tussen iedere twee knopen, is de tussencentraliteit van een knoop simpelweg gelijk aan het aantal kortste wegen die via hem lopen.



Figuur 3

Een P2P-netwerk met vier *peers* met hun voltooide data-overdrachten in MegaByte. De maximale datastroom van *peer* A naar *peer* B is 250, en in omgekeerde richting is die 350 (200 via *peer* C en 150 via *peer* D), zodat *peer* B een goede reputatie bij *peer* A zal hebben.

Binnen het Nederlandse wegen- of spoorwegennetwerk zal Utrecht een grote tussencentraliteit hebben, en dat is dus een goede plaats om de verkeersstromen in de gaten te houden. In Figuur 1 (rechts) is de tussencentraliteit van de centrale knoop gelijk aan 56. Door nu een *peer* reputaties te laten berekenen op dezelfde manier als hiervoor, met als uitgangspunt de *peer* in het netwerk met de hoogste tussencentraliteit in plaats van zichzelf, blijken vaak betere waarden voor de reputaties bepaald te kunnen worden [11].

In zijn algemeenheid moet de decentralisatie van P2P-systemen en de directe communicatie tussen *peers* wel een beetje genuanceerd worden: de communicatie verloopt via *Internet Service Providers* en die kunnen de onderlinge communicatie in de gaten houden en onder invloed van de overheid staan. Dit roept de vraag op wat we precies onder ‘decentralisatie’ moeten verstaan.

Intermezzo: decentralisatie – what's in a word?

Met het huidige internet is het vermoedelijk vrijwel onmogelijk om twee computers in de wereld aan te wijzen die niet met elkaar kunnen communiceren. Vrijwel geen computer werkt nog opzichzelfstaand, want gedistribueerde informatieverwerking is de norm geworden. Hoewel het concept van 'gedistribueerd (computer)systeem' voor de hand liggend lijkt, is het toch goed ons er rekenschap van te geven wat dat precies inhoudt. In mijn colleges op dit gebied definieer ik een gedistribueerd systeem als een systeem dat gekenmerkt wordt door de volgende drie eigenschappen:

1. *Autonomie*, ofwel distributie van autoriteit.
2. *Samenwerking*, ofwel distributie van functionaliteit.
3. *Communicatie*, ofwel distributie van informatie.

Deze definitie refereert op geen enkele wijze aan computers of netwerken, maar is van toepassing is op allerlei typen systemen, met name ook op organisaties zoals bijvoorbeeld bedrijven, universiteiten, sportverenigingen, of simpelweg groepen mensen zoals families. De kracht van dit aspect is dat allerlei problemen die zich in gedistribueerde computersystemen voordoen, zich ook al voordeden voordat computers hun intrede deden en dat we kunnen leren van de oplossingen die voor deze problemen al lang geleden zijn bedacht. Denk hierbij bijvoorbeeld aan het probleem van de vertraging of het verloren gaan van berichten en het gebruik van *time outs* om daarmee om te gaan: als bij gegevensoverdracht tussen twee computers de gegevens lang op zich laten wachten, kan de ontvangende computer de gegevens nogmaals opvragen, of op zijn minst aan de verzendende computer vragen of hij nog in leven is. Ik hoef u niet te vertellen dat *time outs* in het dagelijks leven een belangrijke rol spelen.

In de literatuur over gedistribueerde computersystemen worden de termen 'decentralisatie' en vooral 'gedecentraliseerd' regelmatig gebruikt, maar eigenlijk zonder dat van deze begrippen een definitie wordt gegeven. Daarom ben ik eens gaan kijken naar andere wetenschappen waarin dit wel gedaan is. Je komt dan al gauw terecht in het domein van het openbaar bestuur en de staatsinrichting. De drie dimensies van decentralisatie die daar gewoonlijk worden onderscheiden [18] zijn:

1. *Fiscale decentralisatie*, die aangeeft in hoeverre lagere overheden zelf belasting heffen.
2. *Bestuurlijke decentralisatie*, die aangeeft in hoeverre de overheidsbureaucratie over verschillende niveaus is gespreid.
3. *Politieke decentralisatie*, die aangeeft in hoeverre op verschillende niveaus belangenbehartiging en de daaraan gerelateerde besluitvorming plaatsvindt.

Er is op basis van de gegevens van vele landen, zoals het percentage belasting dat door lagere overheden wordt geheven en het bestaan van verkiezingen op lagere niveaus, nagegaan [18] in hoeverre landen langs de bovengenoemde drie dimensies gedecentraliseerd zijn. In Tabel 1 staat de mate van decentralisatie van een aantal landen genormaliseerd op een schaal van 0 (volledig gecentraliseerd) tot 1 (volledig gedecentraliseerd). Conclusies die aan de tabel ontleend kunnen worden, laat ik aan de lezer. Het zou de moeite waard zijn om een definitie van gedecentraliseerde computersystemen te hebben, of liever zelfs een maat voor de decentralisatie van zo'n systeem. Wat zou zo'n maat behelzen? Welke dimensies zouden we moeten onderscheiden?

Tabel 1. De mate van decentralisatie in drie dimensies van een aantal landen op een schaal van 0 (volledig gecentraliseerd) tot 1 (volledig gedecentraliseerd) [18].

	Mate van decentralisatie		
	Fiscaal	Bestuurlijk	Politiek
Canada	0,96	0,67	0,82
Denemarken	0,71	0,53	0,87
Duitsland	0,66	0,64	0,88
Frankrijk	0,29	0,63	0,80
Nederland	**0,45**	**0,20**	**0,44**
VS	0,80	0,56	0,84

Een interessante vorm van decentralisatie is de zogenaamde 'stille decentralisatie' [7], waarbij decentralisatie niet actief wordt nagestreefd, maar onbedoeld tot stand komt door veranderingen in een systeem. Twee van dergelijke veranderingen zijn een verandering in de beschikbaarheid van *resources* en het veranderde belang van beleidsterreinen. De eerste vorm treedt bijvoorbeeld op wanneer een land of een provincie veel welvarender wordt door het vinden van

grondstoffen of door de vestiging van een groot bedrijf. Deze vorm van stille decentralisatie valt ook te betrekken op de informatica. Per slot van rekening doen we in computersystemen niet veel anders dan het beheren van *resources*, zoals rekenkracht, opslagruimte en netwerkcapaciteit. Hierboven haalde ik het voorbeeld aan van de capaciteitsvergroting van de internetverbindingen van de computers van thuisgebruikers. Het gevolg hiervan was een decentralisatie van het internet, met meer mogelijkheden voor de individuele gebruiker.

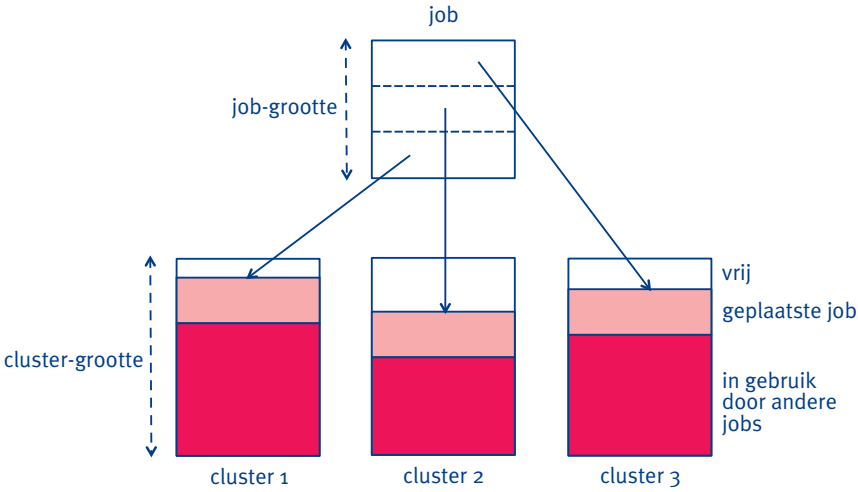
Een voorbeeld van de tweede vorm van stille decentralisatie – een veranderd belang van beleidsterreinen – is dat oorspronkelijk Canada een meer gecentraliseerde federatie vormde dan de Verenigde Staten, maar dat het nu minder gecentraliseerd is. De reden hiervoor ligt erin dat sociaal en economisch beleid in Canada aan de provincies is overgelaten, maar dat deze beleidsterreinen in de 20e eeuw sterk aan belang hebben gewonnen [7]. Een goede parallel van deze vorm van stille decentralisatie in de informatica heb ik niet kunnen vinden.

Een oefening in decentralisatie: processor-coallocatie

Aan de hand van een voorbeeld wil ik laten zien wat voor onderzoek ik doe in mijn tweede onderzoeksgebied, de *scheduling* van *jobs* in grootschalige gedistribueerde systemen. Dit voorbeeld heeft twee ingrediënten die in het dagelijks leven ook voorkomen. Ten eerste is het vaak beter een grote opdracht van wat voor soort dan ook te laten uitvoeren door een groep van personen in plaats van door één enkele persoon. Hetzelfde doen we in de informatica: computerapplicaties, bijvoorbeeld om weersvoorspellingen te doen of om stromingen in water na te bootsen, worden geparallelliseerd zodat ze door meerdere processoren in een *cluster* van processoren uitgevoerd kunnen worden om sneller voltooid te zijn. Zo'n *cluster* staat meestal onder het beheer van een enkele *scheduler* die volledig bepaalt wat iedere processor doet. Vaak gaat parallellisering wel ten koste van de efficiëntie omdat door de benodigde communicatie tussen de processoren de totale tijd die de processoren bij elkaar opgeteld met een applicatie bezig zijn, groter is dan bij het gebruik van een enkele processor.

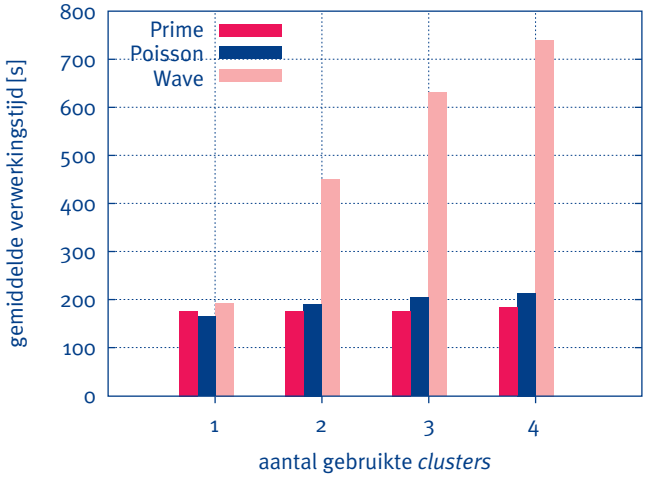
Ten tweede kan het zijn dat de totale groep personen die de opdracht moet uitvoeren uit verschillende deelgroepen bestaat, ieder op zijn eigen locatie en met zijn eigen voorman. Dit leidt tot twee problemen. Allereerst moeten de voormannen het – decentraal – eens worden over de periode waarin de opdracht wordt uitgevoerd. Velen van u hebben wel eens ervaren hoe moeilijk het is om met een aantal mensen een afspraak te maken als iedereen autonoom is in het beheren van zijn agenda. In computersystemen vertaalt zich dit in coördinatie tussen de *schedulers* van verschillende *clusters* om processoren uit die *clusters* gelijktijdig toe te wijzen aan een applicatie. We noemen dit processor-coallocatie. Verder kan, doordat nu ook communicatie over grotere afstand tussen de *clusters* nodig is, de efficiëntie nog verder afnemen. In feite biedt processor-coallocatie de snellere beschikbaarheid van meer processoren in verschillende *clusters* samen tegen een lagere efficiëntie. Figuur 4 toont een voorbeeld van de coallocatie van een *job* die alleen op een systeem bestaande uit drie *clusters* kan worden geplaatst door hem in meerdere stukken te splitsen. Figuur 5 laat de invloed van de lagere efficiëntie van processor-coallocatie op de verwerkingstijd van drie applicaties zien. De *Wave*-applicatie heeft blijkbaar veel te lijden van de langzamere *wide-area*

communicatie. Een bijkomend aspect is nog of de *jobs* van tevoren al in stukken zijn opgesplitst die ieder in hun geheel in één *cluster* moeten draaien, of dat de *scheduler* de *jobs* naar believen zelf mag opsplitsen.



Figuur 4

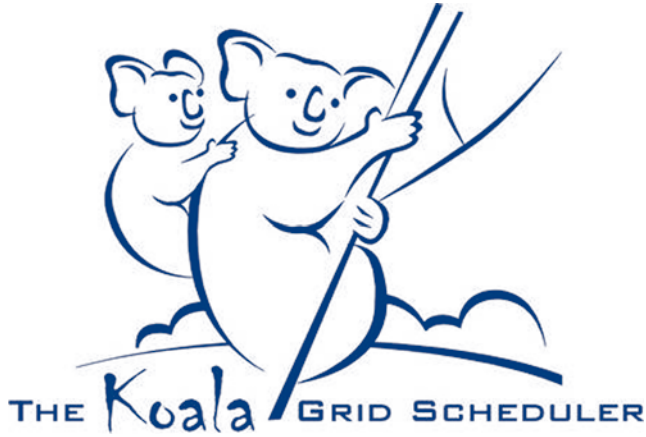
Een voorbeeld van processor-coallocatie waarbij een *job* die niet in zijn geheel op de vrije processoren in een enkel *cluster* past, geplaatst wordt door hem te spreiden over drie *clusters*. De *job*-grootte en de *cluster*-grootte worden uitgedrukt in aantal processoren.



Figuur 5

De gemiddelde verwerkingstijd van een drietal applicaties bij het gebruik van processor-coallocatie met in totaal 32 processoren in 1, 2, 3, en 4 *clusters*.

Over een periode van een aantal jaar heb ik met drie promovendi onderzoek gedaan naar processor-coallocatie. We zijn begonnen met het simuleren van allerlei *scheduling policies* [5], bijvoorbeeld de *policy* die van tevoren opgesplitste *jobs* zodanig spreidt dat de belasting in de *clusters* evenwichtig verdeeld is, of de *policy* die de *jobs* zodanig opsplitst dat ze over zo min mogelijk *clusters* worden gespreid. Daarbij kwam ook enige wiskundige analyse te pas om na te gaan hoeveel capaciteit er verloren gaat doordat er altijd 'gaten' van beschikbare processoren overblijven waar geen enkele *job* in past. Later hebben we een *multicluster scheduler* ontwikkeld [16] voor werkelijke systemen die processor-coallocatie ondersteunt. Hiermee hebben we veel experimenten gedaan [19]. Deze scheduler hebben we KOALA genoemd. Dit is niet een of andere ingewikkelde afkorting, maar een naam die in uitspraak enigszins lijkt op coallocatie en leuke plaatjes oplevert, zie Figuur 6. Helaas ontdekte ik een paar jaar geleden, tijdens een reis naar Australië, dat koala's erg lui en dom zijn. Overigens is het kiezen van een aansprekende naam voor onderzoeksprojecten tegenwoordig erg belangrijk, men onthoudt ze, en ze leveren daardoor grotere zichtbaarheid en dus wellicht meer citaties op! *Marketing*, *branding* en *advertizing* zijn belangrijke activiteiten voor een onderzoeker geworden.



Figuur 6

Het logo van de KOALA *multicluster scheduler*.

Een van de belangrijkste bevindingen van ons onderzoek naar processor-coallocatie is, dat het alleen de moeite loont om het te gebruiken in een systeem waaraan veel *jobs* worden aangeboden als de verwerkingstijd van de *jobs* door coallocatie met minder dan 20% stijgt.

We hebben KOALA uitgebreid voor allerlei andere typen applicaties, waarbij het idee van spreiding over meerdere *clusters* steeds centraal is blijven staan. Overigens doen we dat nog steeds, op dit moment met een promovendus aan zowel de TU Delft als aan de TU/e, voor typen applicaties die ook veel in *data centers* worden gebruikt. In een meer fundamentele studie van een decentraal mechanisme voor de spreiding van computerapplicaties over autonome *clusters* van computers, hebben we al meer dan vijftien jaar geleden het zogenaamde Condor *flocking*-mechanisme ontworpen [8].

Er zijn twee redenen waarom het soort onderzoek dat ik zojuist beschreven heb, mij na aan het hart ligt. Ten eerste leidt een zich aan echte systemen ontleend probleem zowel tot praktische als tot meer fundamentele onderzoeksvragen en -antwoorden. Coallocatie is in feite een algemenere, meerdimensionale vorm van *bin packing* – het plaatsen van rechthoekige pakketjes in zo weinig mogelijke rechthoekige bakken. Ook vele jaren vanaf nu, als computers danig veranderd zullen zijn, zijn de betreffende resultaten in hun abstracte vorm nog steeds relevant. Ten tweede zijn de leukste en beste prestatieanalyse-studies die, die alledrie de methoden van prestatie-analyse combineren: wiskundige analyse, simulaties, en metingen – mits de verschillende methoden natuurlijk tot dezelfde resultaten leiden.

Als ik nu het onderzoek in P2P-systemen en *scheduling* dat ik aangestipt heb, karakteriseer, dan kan ik zeggen dat mijn onderzoek zich richt op de fundamentele aspecten van het ontwerp en op de structurele en vooral kwantitatieve analyse van gedistribueerde computersystemen. Daarbij kom ik graag tot de realisatie van een werkelijk systeem om experimenten mee te doen. Dit laatste brengt mij direct tot het volgende intermezzo.

Intermezzo: experimentele informatica

Meer dan alleen theorie, is informatica ook een experimentele wetenschap. Op het eerste gezicht klinkt dit vreemd: we creëren artefacten, namelijk computers en netwerken, en vervolgens doen we alsof dat door de natuur gegeven systemen zijn waarin we grootheden willen meten. Een simpel voorbeeld van zo'n grootheid is de responstijd van opvragingen op het *World Wide Web*. De eerste keer dat op grote schaal dergelijke metingen zijn gedaan, was bij de Olympische Spelen van 1996 in Atlanta [14]. Eenvoudig voor te stellen vragen die men dan probeert te beantwoorden, zijn: Hoeveel computers moeten er gebruikt worden om de verwachte hoeveelheid werk te kunnen afhandelen? Moeten alle computers alle vragen kunnen beantwoorden, of is het beter om de gegevens op te delen over de computers zodat iedere vraag door slechts één of enkele computers kan worden beantwoord? Een extreem voorbeeld van een computersysteem dat wel dichtbij een door de natuur gegeven, organisch gegroeid systeem staat, is het internet.

Er kunnen vier soorten experimenten worden onderscheiden. In *in situ*-experimenten worden echte applicaties op een echt platform uitgevoerd. Hiertoe reken ik ook meetstudies aan operationele systemen, zoals het internet of P2P-systemen. In *emulaties* wordt het echte platform met een versimpeld platform nagebootst, bijvoorbeeld door in een enkel *cluster* van processoren de vertragingen van netwerkpakketten kunstmatig te vergroten om een groter netwerk met meerdere *clusters* na te bootsen. In *benchmarking* worden modellen of componenten van applicaties in echte systemen gedraaid, om bijvoorbeeld supercomputers met elkaar te vergelijken. Tenslotte zijn er *simulaties*, waarin modellen van applicaties op modellen van echte systemen worden uitgevoerd. Simulaties zijn erg flexibel, je kunt er iedere werkelijkheid mee nabootsen of geweld aandoen.

De redenen om in de informatica experimenten te doen, lopen uiteen [9]. Ze kunnen worden gebruikt bij het ontwerp van een nieuw systeem, om na te gaan of het aan de gestelde eisen voldoet. Deze experimenten zijn *engineering*-gedreven en worden veel bij productontwikkeling gedaan. Experimenten kunnen ook een wetenschappelijk karakter hebben als ze worden gebruikt om de prestatie van een nieuw mechanisme te beoordelen, zoals ik heb laten zien bij het onderwerp van processor-coallocatie. Wetenschappelijk zijn ook meetexperimenten die worden

gedaan om operationele systemen te modelleren of hypotheses te testen, zoals de hypothese dat het internet een compleet willekeurige netwerkstructuur heeft, wat niet zo blijkt te zijn. Artikelen met louter meetresultaten worden helaas moeilijk door conferenties of tijdschriften geaccepteerd. Ze kunnen in mijn ogen echter bijdragen aan een goed kwantitatief begrip van computersystemen, wat erg interessant en leerzaam kan zijn – onze artikelen over metingen aan P2P-systemen [15,17] zijn goede voorbeelden – en wat tot goede onderzoeksvragen kan leiden.

Het bedrijven van experimentele informatica gaat met een aantal problemen gepaard. Allereerst zijn er voldoende financiële middelen nodig om de benodigde apparatuur te bekostigen. Nu zijn we in Nederland in de gelukkige omstandigheid dat we een *multicluster*-systeem hebben dat louter voor experimenteel informatica-onderzoek bestemd is, de zogenaamde Distributed ASCI Supercomputer (DAS) [2]. Dit systeem, met *clusters* bij beide universiteiten in Amsterdam, bij de universiteiten in Leiden en Delft, en bij ASTRON, het Nederlands Instituut voor Radio-Astronomie, is voor een groot deel door NWO gesubsidieerd en is nu al in zijn vierde generatie. De verschillende generaties van de DAS hebben veel onderzoek in de informatica in Nederland mogelijk gemaakt en hebben er zeker aan bijgedragen om NWO ervan te overtuigen dat informatica ook een experimentele discipline is, die geld voor apparatuur behoeft.

Ten tweede is er de vraag hoe algemeen geldend de resultaten van experimenten zijn. Resultaten verkregen met verschillende systeem hoeven niet noodzakelijk met elkaar overeen te komen. Ieder systeem is uniek in zijn specifieke structuur en configuratie, en in feite creëren we allemaal onze eigen wereld als we een systeem of simulatiemodel bouwen [9]. Maar als we niet dezelfde conclusies kunnen ontlenen aan waarnemingen in verschillende systemen, bedrijven we dan wel wetenschap? Zijn er wellicht in de informatica verschillende werkelijkheden, in tegenstelling tot in de natuurwetenschappen?

Een derde punt, dat verband houdt met het vorige, is in hoeverre de resultaten van experimenten geëxtrapoleerd kunnen worden naar nog te bouwen systemen of naar veel grotere systemen. Helaas zijn de verschillende generaties van de DAS in aantallen processoren van vrijwel dezelfde omvang gebleven, terwijl de systemen in de industrie veel sterker gegroeid zijn. In Tabel 2 staan gegevens van een aantal machines uit de zogenaamde Top-500 [4] met de 500 krachtigste computers ter wereld, met ter vergelijking de gegevens van de DAS-4. Op het systeem van Amazon kom ik later terug.

Tabel 2. De plaats en de karakteristieken van een paar machines in de Top-500 en van de DAS-4.

Plaats	Land	Leverancier	Machine	Aantal cores
1	VS	IBM	Sequoia	1.572.864
2	Japan	Fujitsu	K computer	705.024
72	VS	zelf gemaakt	Amazon EC2	17.024
500	VS	HP	-	7.236
-	**Nederland**	**Clustervision**	**DAS-4**	**1.584**

Ten slotte lopen we naar mijn mening in het toepassen van goede experimentele methoden in de informatica achter bij de 'echte' bètawetenschappen, zoals de natuurkunde. Om het goed te doen, hoeft er niet zoveel te veranderen, maar zou er meer aandacht moeten worden besteed aan aspecten als:

1. De *exacte beschrijving en verantwoording van experimenten*, zodat anderen in staat zijn ze te reproduceren of althans volledig te begrijpen. Het gros van de experimenten zal natuurlijk nooit worden herhaald vanwege de benodigde investering in tijd en apparatuur of vanwege het gebrek aan toegang tot dezelfde computersystemen. Tot dit aspect behoort ook de validering van de *random number generator* die gebruikt is voor het genereren van allerlei zich op onvoorspelbare momenten voordoende gebeurtenissen die in het onderzochte systeem plaatsvinden.

2. De *vergelijkbaarheid van experimenten*, waarmee ik bedoel dat onderzoekers zich rekenschap geven van ander werk op hetzelfde gebied en experimenten zo inrichten dat hun werk met dat van anderen vergeleken kan worden. Een goede manier om dit te doen is om dezelfde werklasten waaraan systemen onderworpen worden (bijvoorbeeld de stroom *jobs* of *web requests* met hun eigenschappen die aan een *cluster* of *web server* worden aangeboden) te gebruiken. Het creëren en beschikbaar stellen van archieven met dergelijke werklasten ontleend aan bestaande systemen is hiervoor erg belangrijk [12].

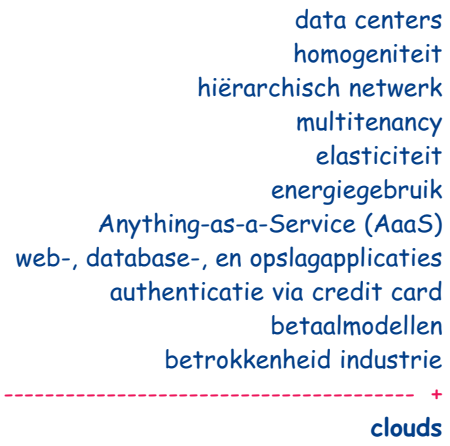
3. De *duidelijke en volledige weergave, analyse en verklaring van de resultaten van experimenten*, inclusief een eventuele foutenanalyse.

Helaas is het met deze aspecten in veel, zowel ter publicatie aangeboden als geaccepteerde artikelen, vaak niet best gesteld. Zo is reproduceerbaarheid vaak onmogelijk doordat de beschrijving van de experimenten niet volledig is, is de kwantitieve analyse van de resultaten vaak gebrekkig en worden de resultaten vaak getoond zonder enige verklaring. Een gebrek aan continue wiskunde die hierbij nodig is in de informatica-opleidingen in Nederland, maar ook in het buitenland, speelt hier zeker een rol.

Voor het onderzoek naar processor-coallocatie hebben we veel experimenten op de DAS gedaan. Zelfs bij dit systeem van bescheiden omvang en betrekkelijk grote homogeniteit is dit bepaald geen sinecure. Er gaat erg veel werk van afstudeerders en promovendi in zitten voordat de experimentele omgeving goed is opgezet. Zelfs dan komt het vaak voor dat van experiment tot experiment de configuratie van de DAS net iets verschilt door storingen in het systeem, zoals het uitvallen van processoren of netwerkverbindingen. Maar het harde werk loont wel!

Coda: clouds

Mijn onderzoek op het gebied van *scheduling* strekt zich tegenwoordig ook uit tot *clouds*, die zo'n zeven jaar geleden hun intrede hebben gedaan. *Clouds* zijn een optelsom van de technologieën en eigenschappen zoals weergegeven in Figuur 7. Ze zijn gebaseerd op *data centers*, grote centrale homogene collecties van computers (denk aan tienduizenden bij internet-giganten als Amazon, Facebook, Google en Microsoft) met snelle netwerken. Meerdere klanten (*tenants*), vaak kleine en middelgrote bedrijven, kunnen tegen betaling aan een *cloud provider* allerlei web- of database-gebaseerde applicaties als dienst installeren en laten uitvoeren in diens *data center*, met een elastisch aantal processoren al naar gelang de vraag van hun eigen klanten. Een belangrijk aspect van *data centers* in het algemeen en *clouds* in het bijzonder is het energiegebruik.



Figuur 7

Clouds als een optelsom van technologieën en eigenschappen.

Een met het oog op universitair onderzoek belangrijk aspect dat in Figuur 7 genoemd wordt, is de betrokkenheid van de industrie, hetgeen in tegenstelling staat tot de situatie die we bij (*computational*) *grids* gewend waren. *Grids* zijn grote, veelal heterogene collecties van computersystemen, die hoofdzakelijk gebruikt worden voor wetenschappelijke applicaties en waarnaar tussen 1995 en

2010 veel onderzoek gedaan is. *Grids* bedienen een nichemarkt die zich beperkt tot de onderzoekswereld. Dientengevolge waren grote computerfabrikanten er niet erg in geïnteresseerd en gingen de ontwikkelingen niet zo snel. Bij *clouds* en *data centers* zijn de grote computerfabrikanten en eigenaars, zoals Amazon en Google, wel zeer betrokken. Het gevolg hiervan is dat als er een goed idee opkomt, deze bedrijven veel onderzoekers en programmeurs aan het werk zetten om het snel te ontwikkelen en in de praktijk te brengen. Universiteiten moeten dus met hun onderzoekscyclus van 4 jaar en een capaciteit per onderzoeksgroep van enkele promovendi een zorgvuldige selectie van onderzoeksvragen maken.

In ons onderzoek naar *clouds* zijn we onder andere hun geschiktheid nagegaan voor wetenschappelijke applicaties [13]. Omdat de infrastructuur er niet voor geoptimaliseerd is, blijkt die geschiktheid nog niet zo best te zijn. Des te opmerkelijker is het dat men erin geslaagd is om een deel van de *cloud* van Amazon zo te configureren dat het hoog in de Top-500 terechtkwam, zie Tabel 2.

Een voor velen verrassend gegeven is dat de bezettingsgraad van *data centers* vaak erg laag is, in de orde van 10-30%. Dit is een gevolg van een erg conservatieve schatting van de hoeveelheid werk die moet worden verricht en het voorbereid zijn op hoge piekbelastingen. Het efficiënte gebruik van *data centers* is dan ook een belangrijke onderzoeksvraag, waar ik mij ook mee bezighoud, onder andere via het dynamisch toewijzen van processoren aan *frameworks* zoals *MapReduce*.

Wat het onderwijs betreft, met ingang van dit najaar geef ik aan de TU/e in de MSc-opleiding Informatica een vak *Cloud Computing*. Dit vak heb ik ontwikkeld in samenwerking met Alexandru Iosup van de TU Delft en het wordt in identieke vorm in Delft en Eindhoven gegeven, met dit jaar in beide plaatsen circa 30 studenten.

Dankwoord

Nu ik aan het einde van deze rede ben gekomen, wil ik enige woorden van dank uitspreken. Ik dank het College van Bestuur van de TU/e en het Bestuur van de Faculteit Wiskunde en Informatica voor het in mij gestelde vertrouwen. Met name dank ik Johan Lukkien, hoogleraar van de groep Systeem Architectuur en Netwerken waar ik nu deel van uitmaak, voor zijn inspanningen om mijn benoeming te realiseren.

Aan de Delftse kant dank ik Henk Sips. Als hoogleraar van de groep Parallelle en Gedistribueerde Systemen (PGS) waartoe ik behoor, heeft hij mij altijd ruim baan gegeven. Tevens heeft hij, ook in zijn rol destijds als plaatsvervangend decaan van de Faculteit Elektrotechniek, Wiskunde en Informatica van de Technische Universiteit Delft, mede mijn benoeming hier aan de TU/e mogelijk gemaakt.

Ten slotte dank ik iedereen met wie ik in de loop van de jaren samen onderzoek heb gedaan en van wie ik daardoor veel heb geleerd. In het bijzonder dank ik Alexandru Iosup en Johan Pouwelse van de PGS-groep van de TU Delft voor de zeer intensieve en vruchtbare samenwerking, die nu al acht jaar duurt. Veel dank ook aan mijn vroegere en huidige promovendi. Door hun vele nationaliteiten bieden zij een erg leerzame omgeving, en ze dwingen me aldoor weer om hard na te denken. Zij vormen de levensader van de universiteit, zeker in een jong en dynamisch vakgebied als de informatica.

Ik heb gezegd.

Referenties

1. Diaspora*, http://diasporaproject.org.
2. De 4e-generatie Distributed ASCI Supercomputer (DAS-4), http://www.cs.vu.nl/das4.
3. Resources on Experimental Computer Science, http://www.cs.huji.ac.il/~feit/exp/related.html.
4. De Top-500, http://www.top500.org.
5. A.I.D. Bucur en D.H.J. Epema, "Scheduling Policies for Processor Co-Allocation in Multicluster Systems," *IEEE Trans. on Parallel and Distributed Systems*, Vol. 18, pp. 958-972, 2007.
6. R. Delaviz, N. Andrade, J.A. Pouwelse, en D.H.J. Epema, "SybilRes: A Sybil-Resilient Flow-Based Decentralized Reputation Mechanism," *32nd Int'l Conf. on Distributed Computing Systems* (ICDCS), pp. 203-213, 2012.
7. H.F.W. Dubois en G. Fattore, "Definitions and Typologies in Public Administration Research: The Case of Decentralization," *Int'l Journal of Public Administration*, Vol. 32, pp. 704-727, 2009.
8. D.H.J. Epema, M. Livny, R. van Dantzig, X. Evers, en J. Pruyne, "A Worldwide Flock of Condors: Load Sharing among Workstation Clusters," *Future Generation Computer Systems*, Vol. 12, pp. 53-65, 1996.
9. D. Feitelson, *Experimental Computer Science: The Need for a Cultural Change*, manuscript available from [3].
10. S.L. Feld, "Why Your Friends Have More Friends than You Do," *American Journal of Sociology*, Vol. 96, pp. 1464-1477, 1991.
11. D. Gkorou, J.A. Pouwelse, en D.H.J. Epema, "Betweenness Centrality Approximations for an Internet Deployed P2P Reputation System," *HotP2P*, pp. 1627-1634, 2011.
12. A. Iosup, H. Li, M. Jan, S. Anoep, C. Dumitrescu, L. Wolters, en D.H.J. Epema, "The Grids Workloads Archive," *Future Generation Computer Systems*, Vol. 24, pp. 672-686, 2008 (see also http://gwa.ewi.tudelft.nl).
13. A. Iosup, S. Osterman, M.N. Yigitbasi, R. Prodan, Th. Fahringer, en D.H.J. Epema, "Performance Analysis of Cloud Computing Services for Many-Tasks Scientific Computing," *IEEE Trans. on Parallel and Distributed Systems*, Vol. 22, pp. 931-945, 2011.

14. A. Iyengar, M.S. Squillante, en Li Zhang, “Analysis and Characterization of Large-Scale Web Server Access Patterns and Performance,” *World Wide Web*, Vol. 2, pp. 85-100, 1999.
15. M. Meulpolder, L. D’Acunto, M. Capotă, M. Wojciechowski, J.A. Pouwelse, D.H.J. Epema, en H.J. Sips, “Public and Private BitTorrent Communities: A Measurement Study,” *9th Int’l Workshop on Peer-To-Peer Systems* (IPTPS), 2010.
16. H.H. Mohamed en D.H.J. Epema, “KOALA: A Co-Allocating Grid Scheduler,“ *Concurrency and Computation: Practice and Experience*, Vol. 20, pp. 1851-1876, 2008.
17. J.A. Pouwelse, P. Garbacki, D.H.J. Epema, en H.J. Sips, “A Measurement Study of the BitTorrent Peer-to-Peer File-Sharing System,” *4th Int’l Workshop on Peer-To-Peer Systems* (IPTPS), Springer LNCS 3640, pp. 205-216, 2005.
18. A. Schneider, “Decentralization: Conceptualization and Measurement,” *Studies in Comparative International Development*, Vol. 38, pp. 32-56, 2003.
19. O.O. Sonmez, H.H. Mohamed, en D.H.J. Epema, “On the Benefit of Processor Co-Allocation in Multicluster Grid Systems,” *IEEE Trans. on Parallel and Distributed Systems*, Vol. 21, pp. 778-789, 2009.

Curriculum Vitae

Prof.dr.ir. Dick Epema is per 1 september 2011 benoemd tot deeltijdhoogleraar Decentralized Distributed Systems aan de faculteit Wiskunde & Informatica van de Technische Universiteit Eindhoven (TU/e).

Dick Epema (1956) studeerde Wiskunde aan de Universiteit Leiden en promoveerde daar in 1983 op een onderwerp uit de algebraïsche meetkunde. Halverwege zijn promotietijd, in 1981, werd Informatica in Nederland een zelfstandige opleiding, en in 1984 werd hij Universitair Docent Technische Informatica aan de toenmalige Technische Hogeschool Delft bij de groep Operating Systems. Daarnaast voltooide hij ook de studie Technische Informatica in Delft. Sinds 1999 is hij Universitair Hoofddocent bij de groep Parallelle en Gedistribueerde Systemen van de TU Delft. Gedurende het academisch jaar 1987-1988, in 1991 en in 1998 was hij *visiting scientist* bij het IBM T.J. Watson Research Center in New York. In 2009 was hij op sabbatical bij de University of California in Santa Barbara.
Zijn onderzoeksgebieden zijn gedistribueerde systemen, meer in het bijzonder *grids*, *clouds* en *peer-to-peer* systemen, en prestatie-analyse van computersystemen, met name *scheduling*. In zijn onderzoek combineert hij wiskundige analyse en simulaties met het ontwerp en de implementatie van echte systemen en experimenten daarmee. Naast zijn eigen onderzoek draagt hij als *General Chair* en *Program Chair* van een reeks van vooraanstaande conferenties bij aan zijn vakgebied.

Colofon

Productie
Communicatie Expertise
Centrum TU/e

Fotografie cover
Rob Stork, Eindhoven

Ontwerp
Grefo Prepress,
Sint-Oedenrode

Druk
Drukkerij Snep, Eindhoven

ISBN 978-90-386-3283-4
NUR 980

Digitale versie:
www.tue.nl/bib/