Methodology

In de wetenschappelijke wereld, bij banken en bij ingenieursbureau's heeft men de grafische processor ontdekt als razendsnelle aanvulling op de gewone CPU. Die prestaties krijg je echter niet cadeau. Je zult je algoritme op laag niveau naar de GPU moeten porten. Gelukkig is er met OpenCL een standaard programeertaal voor deze markt in de maak.

Razendsnel rekenwerk

High-performance met een grafische processor

Grootverbruikers van rekenkracht hebben de afgelopen twee jaar de grafische processor ontdekt als snelle en goedkope bron van verwerkingscapaciteit. Denk daarbij met name aan de GPU's van nVidia en ATI (tegenwoordig onderdeel van AMD), en aan de Cell-processor van IBM. Ontwikkelaars die hun code naar een van deze architecturen porten, melden speed-ups van één of twee orden van grootte ten opzichte van de traditionele CPU's.

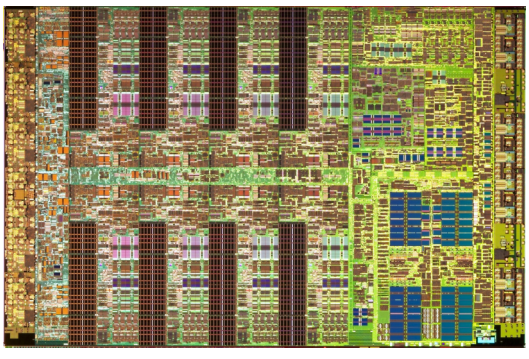
Die versnelling krijg je echter niet cadeau. Grootste problemen bij het porten van een applicatie zijn de mapping van het algoritme op de architectuur van de GPU en het uitnutten van alle beschikbare parallelliteit. Dat betekent dat het algoritme om te beginnen al voldoende mogelijkheden in zich moet hebben om te worden geparallelliseerd. Daarbij gaat het niet om het draaien van een handvol high-level threads zoals we dat tegenwoordig op een multicore processor doen, maar om het creëren van duizenden of tienduizenden mini-threads die allemaal tegelijk dezelfde berekening uitvoeren op een klein stukje data.

General-Purpose Computing op een GPU, kortweg GPGPU genoemd, vraagt dan ook om een hele andere manier van programmeren. Grafische processoren werden immers nooit ontwikkeld om algemene rekenklussen uit te voeren. Ze zijn volledig geoptimaliseerd voor het verwerken van graphics. Denk dan met name aan massief parallelle berekeningen op grafische datastructuren als arrays, pixels, kleuren, paletten en textures. Dat betekent dat het uitnutten van deze hardware behoorlijk wat ouderwets handwerk van de programmeur vraagt, en dat alleen een heel specifieke klasse van algoritmen goed op deze manier te parallelliseren is.

Waar op gewone processoren de control flow centraal staat, dat wil zeggen de manier waarop het algoritme data en hulpvariabelen verwerkt, zijn GPU's gebaseerd op het zogenaamde stream processing model. Daarbij is een belangrijke rol weggelegd voor matrices en andere grote datastructuren waar razendsnel en in parallel, relatief eenvoudige bewerkingen op de afzonderlijke elementen worden uitgevoerd.

Persoonlijke super-computers

Grafische processoren hebben dan ook veel grotere lokale geheugens en registerbanken dan traditionele CPU's. Dat maakt ze bij uitstek geschikt voor massief parallelle applicaties zoals die in de wetenschap (HPC: High-Performance Computing), de financiële wereld (Monte Carlo berekeningen), en door ingenieursbureaus en de industrie (HPTC: High-Performance Technical Computing) worden gebruikt. Inmiddels worden deze "persoonlijke super-computers" niet alleen door gespecialiseerde leveranciers aangeboden, maar ook door HP, Lenovo en Dell.
Gezien de winstmarges die in de HPC en HPTC wereld worden gemaakt, is het verbazingwekkend hoe traag GPU-leveranciers op deze ontwikkeling ingesprongen zijn. Zo hebben double-precision floating-point berekeningen inmiddels wel hun intre-



De IBM 1001

Aad Offerman
is freelance redacteur

de gedaan, maar zijn de prestaties van de GPU's op dat gebied nog bedroevend.
Hetzelfde geldt voor de toegang tot het geheugen. Grafische processoren presteren het best bij achtereenvolgende, relatief eenvoudige, massief parallelle bewerkingen op grote matrices, met zo min mogelijk geheugen-transacties en zo min mogelijk control flow instructies als lussen (loops) en conditionele sprong-opdrachten (conditional branches).

CUDA

Desondanks heeft zowel nVidia als ATI inmiddels een productlijn speciaal voor GPGPU-toepassingen beschikbaar. Voor ATI zijn dat de Stream-producten (voorheen Close-To-Metal, CTM). Voor nVidia zijn dat de Tesla-kaarten, gebaseerd op hun GeForce 8 GPU's. Deze laatste zijn te gebruiken via de CUDA library (Compute Unified Device Architecture), die inmiddels als de de facto standaard fungeert.
CUDA is beschikbaar voor alle drie productlijnen van nVidia: GeForge voor consumenten, Quadro voor visualisatie en Tesla voor HPC. Die laatste kaarten zijn voorzien van grotere frame buffers voor berekeningen op grote data sets. Op dit moment worden C/C++, DirectX 11 (DirectCompute library) en OpenGL ondersteund. Binnenkort komt daar Fortran bij.
Hier in Nederland is de Amsterdamse VU bezig met het organiseren van van een college CUDA. Hoewel dat nog in een pril stadium is, wil men uiteindelijk een "nVidia center of excellence" worden. In België heeft de Universiteit Antwerpen vergelijkbare ambities.
Commerciële toepassingen worden meestal niet publiek gemaakt. Er is wel een case-beschrijving van BNP Paribas. "Maar ook ABN-AMRO en ING gebruiken GPGPU voor hun risico-calculaties. ABN-AMRO port zijn modellerings-tools naar GPU." Volgens Andrew Humber, senior pr manager voor de Tesla-producten, liggen de meeste toepassingen in de financiële sector, de olie- en gasindustrie, en de medische wereld. "Maar de meeste klanten willen daar niet over praten. Dit is nog een hele jonge technologie, dus erg concurrentiegevoelig."

Cell-processor

IBM zit met zijn Cell-processor tussen de klassieke CPU en de GPU in. De Cell Broadband Engine (CBE) zoals hij officieel heet, is speciaal voor de PlayStation 3 ontwikkeld in samenwerking met Sony en Toshiba (STI). Toen zij eenmaal de mogelijkheden van deze processor voor andere toepassingen hadden ontdekt, begonnen onderzoekers aan universiteiten die game consoles op te stapelen voor hun parallelle rekenwerk.

Peter Hofstee, werkzaam bij IBM als hoofdarchitect van de Cell-rekenkernen, verzekert ons echter dat de processor al vanaf het begin ontworpen werd voor een brede inzetbaarheid. "We waren op zoek naar technologie die breder inzetbaar was. Het belangrijkste doel was efficiëntie, in samenhang met programmeerbaarheid en toepasbaarheid.


Peter Hofstee: Op zoek naar technologie die breder inzetbaar is.

Dure illusie

Bij het ontwerp van de Cell is volgens Hofstee vooral gekeken naar de grootste bron van inefficiëntie: de toegang tot het geheugen. "Het DRAM geheugen zit heel ver weg van de rekenkernen. Dat geldt voor alle general-purpose processoren, die van onszelf, die van Intel en die van AMD. Als je naar de plattegrond van een moderne microprocessor kijkt, dan moet je moeite doen om de optellers en vermenigvuldigers te vinden. Tweederde van de chip, als het niet meer is, wordt in beslag genomen door de cache. Die gebruiken we om de programmeur het idee te geven dat het geheugen vlakbij is en oneindig groot. Maar dat is een hele dure illusie."

Voor de architectuur van de Cell heeft IBM besloten de transparantie van het geheugen voor de programmeur los te laten. De processor bevat naast de centrale kern gebaseerd op de Power-processor acht zogenaamde Synergistic Processing Elements (SPE's). Deze kunnen tegelijkertijd aan het werk worden gezet voor parallelle rekenklussen. Het centrale geheugen is echter niet zomaar vanuit die rekenkernen te benaderen. Programmeurs moeten voor elke taak expliciet aangeven welke variabelen ze in het lokale geheugen willen laden en welke ze daarna weer willen wegschrijven. Die "boodschappenlijstjes" maken de Cell een lastiger processor om voor te coderen.

'Boodschappenlijstjes' maken de Cell lastig om voor te coderen

“We willen weten hoeveel TFLOPS je haalt per Watt”

“Voordat je met het eigenlijke rekenwerk aan de slag kunt, moet je een lijstje maken van alle “ingrediënten” die je nodig hebt. Die gegevens breng je vervolgens over vanuit het hoofdgeheugen naar de local store van de rekenkern. Na het uitvoeren van je programma doe je het omgekeerde: je maakt een lijstje van de zaken die weer terug naar het hoofdgeheugen moeten. Dat is een fundamenteel andere manier om met de traagheid van geheugen-transacties om te gaan.”

Dat maakte de Cell wel moeilijk toegankelijk voor een brede toepasbaarheid. “Zoals met een heleboel ontwikkelingen in de computerwetenschap krijg je dan een situatie waarin universiteiten en HPC-gebruikers als eerste instappen. Als je een heel groot computersysteem hebt, dan kun je je veroorloven om extra energie in de programmering ervan te steken. Op dit moment werken we bij IBM aan het breder toegankelijk maken van de Cell-processor.”

Software-telescoop

Waar je voorheen enorme schotels moest bouwen om gedetailleerde beelden uit de ruimte te kunnen ontvangen, is de laatste telescoop juist gebaseerd op een heleboel kleine antenne’s. LOFAR (Low Frequency Array), een project van ASTRON (Astronomisch Onderzoek in Nederland), bestaat uit zevenduizend losse antenne’s die in een gebied rond Exloo (in Drenthe) staan opgesteld. Ze vormen tesamen vijf spiraalvormige armen. Hier in Nederland beslaat deze radiotelescoop een gebied met een straal van honderd kilometer. Maar er staan ook antennestations in Duitsland, Engeland, Frankrijk en Zweden.

LOFAR moet zicht geven op het begin van ons heelal. Daarvoor is een telescoop nodig die honderd maal gevoeliger is dan wat we al hadden. In schotelvorm zou je die absoluut niet kunnen bouwen, al was het alleen maar vanwege de draaibare ophanging. LOFAR levert die hoge nauwkeurigheid tegen hele lage kosten. Daarbij wordt gebruikt gemaakt van de draaiing van de aarde om het beeld compleet te maken. Richten is niet nodig; er wordt in één keer tegelijkertijd naar de hele hemel gekeken.

200 Gbps

Ondanks de lage prijs van de installatie zelf, krijg je die enorme nauwkeurigheid toch niet cadeau. Er moet namelijk enorm hard gerekend worden aan alle binnenkomende signalen. “We bestrijken 248 subbanden, die elk weer onderverdeeld zijn in 256 subkanalen,” vertelt onderzoeker Rob van Nieuwpoort. “Daarop wordt 768 maal per seconde een meting gedaan. Die 16 bit samples worden ter plekke op de antenne-stations door FPGA’s (Field-Programmable Gate Array) omgezet naar 32 bit single-precision.” En die output wordt vervolgens per glasvezel verzameld en naar Groningen gestuurd.

Rob van Nieuwpoort

Voor het combineren van alle metingen, ook nog eens over de tijd, is een zware software-pijplijn opgezet. “Met deze telescoop verplaats je alles naar de software.” De achterliggende IBM Blue Gene krijgt zo in totaal 200 Gigabit per seconde te verwerken. Daarmee is LOFAR een van de grootste data-verwerker ter wereld. Ter vergelijking: de LHC deeltjesversneller van het CERN levert 300 Mbps aan rauwe data op.

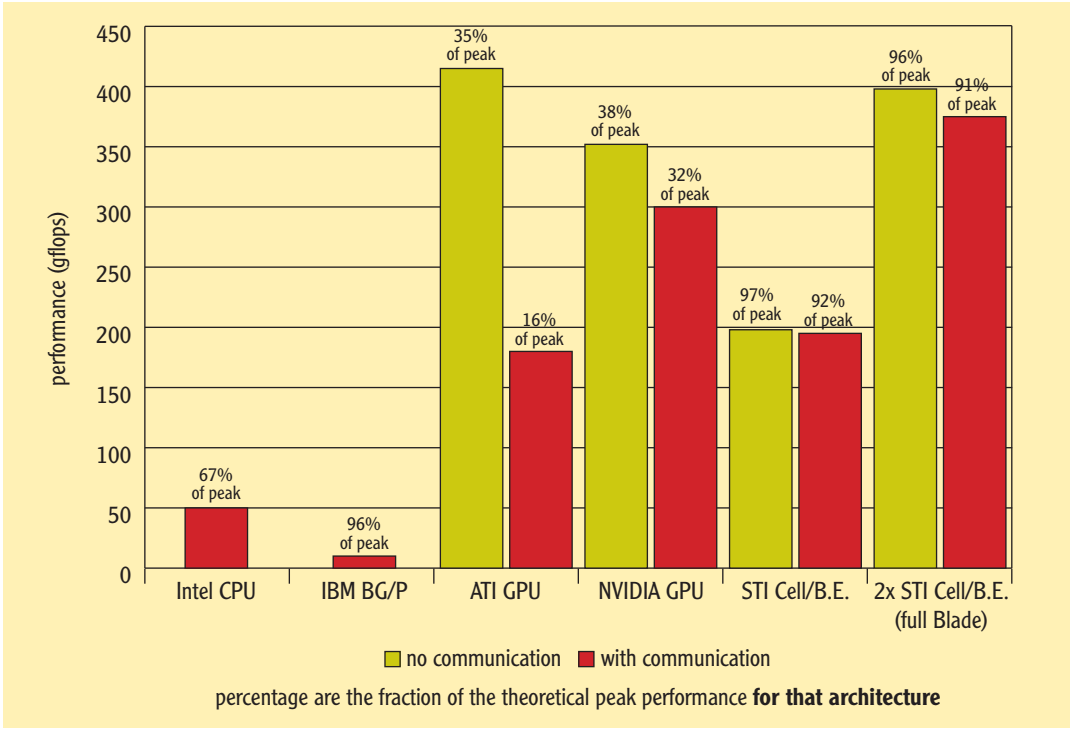
ASTRON is een van de onderzoeksinstituten die kijkt naar de mogelijkheden van GPGPU. De software voor hun LOFAR-telescoop (zie kader) is bij uitstek geschikt om op een grafische processor te draaien. Onderzoeker Rob van Nieuwpoort heeft het correlatie-algoritme voor LOFAR op verschillende hardware-platforms getest en de resultaten naast elkaar gezet. Referentie daarvoor is de huidige IBM Blue Gene computer die Astron nu in Groningen heeft staan voor zijn LOFAR-pijplijn. “Die heeft een capaciteit van 42 TFLOPS (duizend miljard Floating-Point Operations per Second).” Een enkele ATI-processor heeft een piekvermogen van 1,2 TFLOPS. In theorie zou je dus aan dertig van die processoren genoeg hebben om de capaciteit van de huidige Blue Gene te evenaren. “Er zitten twee van die chips op één 4870 PCI-kaartje. Dat levert 2,4 TFLOPS per PCI slot op. De nieuwe 5970-kaart haalt zelfs 4,6 TFLOPS. Daarmee zou je in theorie maar tien kaarten nodig hebben om de rekencapaciteit van onze Blue Gene te evenaren.”

De praktijk is natuurlijk weerbarstiger dan dit eenvoudige rekensommetje suggereert. “Belangrijke vraag is of je dat piekvermogen inderdaad haalt. Onze berekening is vreselijk simpel: alleen maar vermenigvuldigen en optellen. Maar je moet al die data er wel in en uit krijgen. Bovendien willen we weten wat het stroomverbruik is, dus hoeveel TFLOPS haal je per Watt.”

Piekvermogen

De afgelopen twee jaar heeft Van Nieuwpoort vijf verschillende platforms met elkaar vergeleken: de huidige Blue Gene, de Intel Core i7, de ATI 4870, de nVidia Tesla C1060 en de IBM Cell-processor (enkelvoudig en in duo-configuratie op een blade). Voor elk van daarvan werd het daadwerkelijke vermogen gemeten, eerst zonder geheugen-transacties (waarbij alleen de berekening zelf werd uitgevoerd op dummy data), daarna met geheugen-transacties. Nevenstaande figuur laat de uitkomsten daarvan zien. Daaruit blijkt overduidelijk dat de Power en Cell-systemen geen enkele moeite hebben om hun theoretische piekvermogen te halen. Voor alle drie systemen zit het daadwerkelijke vermogen inclusief de geheugen-transacties boven de negentig procent van het piekvermogen.

Voor de Power is transportcapaciteit in en rond de processor voldoende om de rekenkernen continu aan het werk te houden, zo blijkt uit deze statistieken. Het geheim daarvan is natuurlijk de lage kloksnelheid van deze systemen, waardoor de memory wall tot een minimum beperkt blijft. “Er zitten vier kernen in zo’n Power-processor,” zegt Van Nieuwpoort, “maar die draaien op maar 850 MHz. Voor moderne CPU-begrippen is dat heel langzaam. Maar er zitten er gewoon heel veel in een Blue Gene sys-



Prestaties van de verschillende platforms (als percentages van het theoretische piekvermogen van die architectuur).

teem, 12.480 om precies te zijn. Bovendien bevat zo'n systeem een heleboel speciale netwerken, wel vijf verschillende in totaal. Dus daar komen die grote prestaties vandaan."

TFLOPS per kubieke meter

Het verschil tussen de rekenkernen wordt gelijk duidelijk als je de Blue Gene processoren afzet tegen de Core i7 van Intel. Die laatste haalt maar tweederde van zijn piekvermogen, maar dat is nog steeds een veelvoud van de prestaties van een enkele IBM processor. "Door hun lage kloksnelheid verbruiken ze echter heel weinig stroom," aldus Van Nieuwpoort. "En doordat de koeling naar verhouding simpel blijft, kan IBM die processoren heel dicht op elkaar pakken. Zo haal je dus een heel hoge TFLOPS per kubieke meter."

De twee GPU's halen allebei ongeveer een derde van hun piekvermogen, zonder geheugen-transacties wel te verstaan. Het verschil tussen nVidia en ATI wordt duidelijk als het dataverkeer ook meegenomen wordt. Dan haalt ATI nog maar 14 procent van zijn piekvermogen, terwijl de prestaties van nVidia maar een klein beetje terugzakken. Zo kan het dus gebeuren dat de ATI-processor in eerste instantie harder lijkt te gaan, maar dat nVidia (voor dit algoritme) beter presteert als ook de communicatie bij de berekeningen wordt meegenomen. "De hardware van ATI is heel mooi," zegt Van Nieuwpoort, "die loopt op zich heel hard. Alleen het verplaatsen van de data van je CPU naar je GPU toe over de PCI-bus is echt heel traag. Die bus haalt in theorie acht Gbps, maar in de praktijk zijn dat er maar vijf ofzo. Bovendien is ATI niet goed in double buffering: rekenen aan de ene buffer terwijl je tegelijkertijd de volgende buffer binnenhaalt. nVidia kan dat veel beter; dat scheelt bijna niets." Of dat verschil in de drivers of in de architectuur zit, durft Van Nieuwpoort niet te zeggen.

Onwetenschappelijk

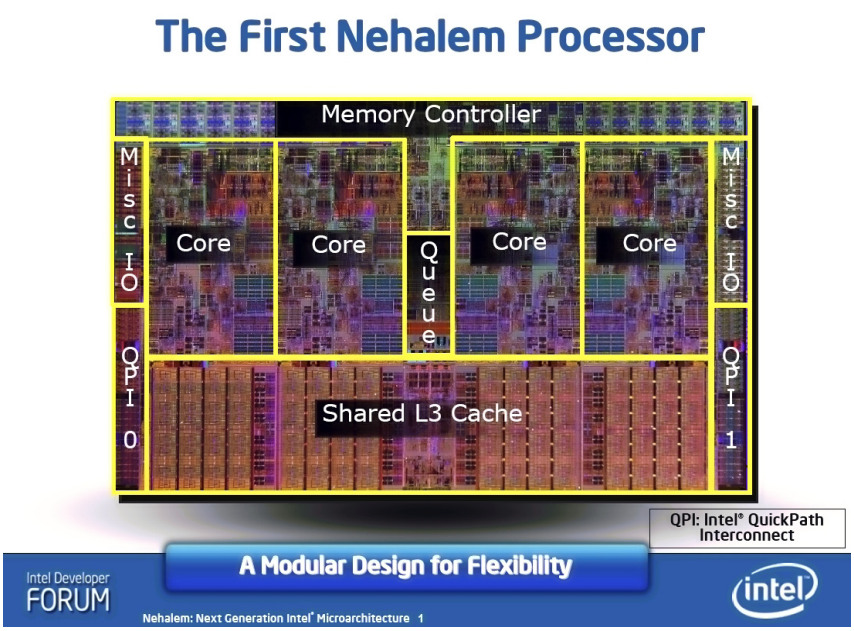
Hoewel de ATI GPU een theoretisch piekvermogen van 1,2 TFLOPS heeft, liggen de daadwerkelijke prestaties van de twee grafische processoren tussen de 200 en 400 GFLOPS. Dat is nog steeds een factor vijf hoger dan die van de Core i7 CPU. "Die veelgehoorde speedup met een factor honderd vind ik onwetenschappelijk," aldus Van Nieuwpoort. "Mensen die zeggen dat hun applicatie op een GPU honderd keer zo hard loopt als op een CPU, gaan allemaal uit van een andere processor. De een neemt een Core i7 als uitgangspunt, de ander een drie jaar oude Opteron." Bovendien gaat men daarbij meestal uit van een niet-geoptimaliseerde executable zoals die door een standaard back-end wordt gegenereerd, zonder SSE.

"Daarom zeg ik ook niet "dat ding gaat vijf keer zo hard". Ik vergelijk ten opzichte van het eigen piekvermogen. Ik zeg "dat ding haalt dertig procent van wat hij in theorie zou kunnen met al die multipliers, adders en andere units die erop zitten. Dat is veel eerlijker."

Assembly

Belangrijkste nadeel van al die verschillende hardware is dat je voor elke architectuur hele andere algoritmen moet schrijven. "De code bestaat voor het grootste gedeelte uit memory management,"

De prestaties worden bereikt door de speciale netwerken en vele cores



"Een groot gedeelte van de moderne microprocessor bestaat uit cache. Die moet de programmeur het idee te geven dat het geheugen vlakbij is en oneindig groot. Maar dat is een hele dure illusie".

Eén programmeertaal waarin je je verschillende algoritmen kunt opschrijven

vertelt Van Nieuwpoort, "en is dus absoluut niet overdraagbaar. Ik heb voor de Cell hele andere algoritmen dan voor de GPU's. Omdat we echt alle performance uit die processoren wilden halen, hebben we alles in assembly geschreven. De tools zijn toch nog niet zo goed dat je alles in C kunt doen. Op de Blue Gene scheelt dat een factor tien. Voor de Cell hebben we een combinatie van C en assembly gemaakt. Je werkt daar met zogenaamde intrinsics, waarbij wel de instructie wordt gespecificeerd maar nog niet de registers die gebruikt worden. Dat zoekt de compiler dan zelf uit, een beetje tussen C en assembly in."

"Dat betekent dus dat je op heel laag niveau met programmeren bezig bent. Je moet de architectuur goed leren kennen, en vervolgens nadenken of wat je precies in C wilt doen en wat in assembly. Meestal doe je de setup in C en de kernen in assembly. En dat moet je dus voor elke architectuur opnieuw doen. Als je echt de laatste performance eruit wilt halen, dan doe je dat niet in een paar weken of maanden. De CUDA toolkit is volgens Van Nieuwpoort een positieve uitzondering. "We hebben de GPU van nVidia zowel in CUDA als in assembly geprogrammeerd. Het verschil in prestaties tussen die twee applicaties was maar iets van tien procent. Voor de Cell en de Blue Gene scheelde dat ongeveer factor tien." De compiler back-ends van IBM zijn dus niet zo goed. "Maar die voor de Cell is de laatste tijd wel beter geworden."

OpenCL

De laatste trend binnen deze op zichzelf al nieuwe ontwikkeling is de opkomst van OpenCL als uniforme programmeertaal voor parallelle toepassingen. Deze taal is door Apple ontwikkeld specifiek voor het uitnutten van grafische (of breder gesteld: heterogene) processoren voor parallelle werklasten. De taal zelf is gebaseerd op OpenGL en OpenAL voor het programmeren van respectievelijk 3D en audio applicaties.

Apple heeft OpenCL inmiddels officieel geïntroduceerd als onderdeel van Mac OS X versie 10.6 (codenaam Snow Leopard). Maar ook IBM, Intel, RapidMind, nVidia en AMD hebben zich inmiddels achter OpenCL geschaard. Voor deze laatste is het zelfs de vervanger van de Brook+/CAL toolkit (het oude CTM, inmiddels onderdeel van de Stream SDK). nVidia heeft voor OpenCL een hele nieuwe software stack gebouwd. Ze gebruiken wel dezelfde intermediate assembly, maar alles daarboven is helemaal onafhankelijk van CUDA."

"Het is echter niet zo dat je in één keer je code in OpenCL schrijft, en dat je dan klaar bent voor zowel de Cell als de GPU's. Je moet voor elke architectuur andere optimalisaties doen. Je hebt dus nog steeds geen taal waarin je één keer je algoritme beschrijft, dat vervolgens goed draait op de Cell, op een GPU en op een multi-core processor. Maar je hebt nu wel één programmeertaal waarin je je verschillende algoritmen kunt opschrijven. Je hoeft niet meer al die toolkits te leren gebruiken."

Factor drie

Hoewel Van Nieuwpoort op dit moment op alle drie architecturen nog een factor drie verschil meet van OpenCL code in vergelijking met native low-level code, verwacht hij dat dit gat in de nabije toekomst zal verdwijnen. "Er is geen reden waarom OpenCL uiteindelijk langzamer zou zijn dan de CUDA toolkit. En die code was maar tien procent langzamer dan handgeschreven assembly. OpenCL kan daar op termijn ook aan komen. Vandaag de dag is OpenCL nog niet te doen, maar conceptueel steekt het goed in elkaar, en de prestaties worden per maand beter. De implementaties die zowel door nVidia als AMD op de laatste Super Computing beurs (SC09) werden gepresenteerd, zouden nog maar een 10-20% penalty hebben ten opzichte van respectievelijk CUDA en Brook+/CAL. Of dat echt zo is, heb ik nog niet kunnen controleren. Er was in ieder geval veel belangstelling vanuit de industrie voor OpenCL." «