Samenvatting

Wetenschappelijke applicaties uit verschillende domeinen (b.v. astronomie, bio-informatica) worden normaliter gemodelleerd als many-task computing (MTC) opdrachten. In dit computing-paradigma drukken wetenschappers en onderzoekers hun complexe data-analyses uit als verzamelingen (zonder sterke relatie) scripts, bags-of-tasks, wetenschappelijke workflows, MapReduce of een combinatie hiervan. MTC bevindt zich op de intersectie van high-performance computing en high-throughput computing omdat deze applicaties over het algemeen worden gevormd door taken zonder sterke relatie die, in tegenstelling tot het message-passing mechanisme dat in high-performance computing wordt gebruikt, via het bestandssysteem communiceren.

Dergelijke applicaties worden gevormd door een groot aantal taken die enerzijds onderlinge data-afhankelijkheden laten zien en anderzijds gebruik maken van communicatiepatronen via bestanden die leiden tot grote hoeveelheden data tijdens de uitvoering. Qua structuur hebben MTC applicaties grote parallelle fasen die gekoppeld zijn door data-aggregatie- en data-opdelingstaken. De eerstgenoemde taken lezen data die door veel andere taken worden gegenereerd terwijl de laatstgenoemde data produceren die door veel andere taken zullen worden gelezen. Dit soort communicatiepatronen leidt vaak tot data-opslag- en netwerkverkeer-onbalans en tot een hoge variabiliteit in het behaalde parallellisme waardoor het beperkt schaalbaar is. Bovendien laten MTC applicaties ook een hoge variabiliteit in het data-gebruik zien.

Eerdere studies hebben laten zien dat traditionele opslagsystemen gebaseerd op harde schijven de performance en schaalbaarheid van MTC applicaties beperken doordat ze niet om kunnen gaan met de mate waarin deze applicaties data genereren. In dit proefschrift onderzoeken we de performance implicaties van het opslaan van data in het werkgeheugen in tegenstelling tot het opslaan op harde schijven. We stellen voor om de applicatie-data in het geheugen van de compute-nodes te plaatsen, geünificeerd in een snel, gedistribueerd bestandssysteem in het werkgeheugen.

Het nadeel van het opslaan van data in het werkgeheugen is dat harde schijven nog steeds over ordes van grootte hogere capaciteit beschikken. Hierdoor kunnen op dit moment alleen applicaties waarvan de data in het werkgeheugen passen voordeel ondervinden van gedistribueerde werkgeheugen-bestandssystemen.

Onze aanpak is tweeledig: applicatie-specifieke ten opzichte van platform-specifieke ontwerpen. Als eerste richten we onze aandacht op het geschikt maken van het gedistribueerde werkgeheugen-bestandssysteem om applicatie-specifieke knelpunten op te heffen: opslag- en netwerkverkeer-onbalans en beperkte schalingsmogelijkheden. Opslag- en netwerk-onbalans vermindert de applicatie-performance terwijl de beperkte schalingsmogelijkheden tot slecht gebruik van de computationele middelen van het onderliggende systeem leiden.

Als tweede gaat onze aandacht uit naar het toespitsen van het werkgeheugen-opslagsysteem op de rekensystemen voor MTC applicaties waarop wij ons richten: privé clusters of publieke cloud systemen. Voor het eerste type systeem geldt dat de netwerk-performance stabiel en voorspelbaar is. Echter, het tweede type systeem heeft te maken met een hoge graad van variabiliteit in netwerk-bandbreedte door co-locatie en virtualisatie overhead die de performance in grote mate ondermijnt. Daarnaast worden huidige rekensystemen tot in grote mate onderbenut wat betreft geheugen- en netwerkgebruik. Studies laten zien dat tot 50% van het geheugen onbenut blijft terwijl er ook een grote hoeveelheid netwerk-bandbreedte beschikbaar is.

Om applicatie-performance te verbeteren, presenteren wij in hoofdstuk 2 het ontwerp van MemFS, een lokaliteits-agnostisch werkgeheugen-opslagsysteem. MemFS spreidt data in gelijke mate over de compute-nodes waarmee het zeer gebalanceerd opslag- en netwerkverkeer bewerkstelligt. Om goede performance te behalen maakt MemFS gebruik van moderne hogesnelheidsinterconnects zoals InfiniBand of 10G Ethernet. Ons systeem verslaat een state-of-the-art lokaliteits-gebaseerd systeem en is toepasbaar in zowel clusters als clouds met hoge kwaliteits netwerken. Dit werk is gepubliceerd in [108, 112].

Hoofdstuk 3 behandelt het ontwerp van MemEFS, een elastisch, gedistribueerd werkgeheugen-bestandssysteem dat zich richt op het volledig benutten van de computationele middelen van het onderliggende systeem. MemEFS slaagt erin de efficiëntie hiervan te verhogen door dynamisch compute-nodes toe te voegen of te verwijderen afhankelijk van de vraag naar opslag door de applicatie. Dit werk is gepubliceerd in [110]. Daarnaast is MemEFS geselecteerd als IEEE TCSC finalist [111] omdat het schaalbaarheid demonstreerde in drie dimensies: horizontaal, verticaal en elastisch.

In hoofdstuk 4 presenteren wij voor MemEFS het ontwerp van een aanpassingsmechanisme voor het netwerk om het netwerkvariabiliteitsprobleem op te lossen. Dit mechanisme neemt de bandbreedtecapaciteit van de compute-nodes waar en herspreidt de data dienovereenkomstig. Dit systeem verbetert de executietijd van MTC applicaties aanzienlijk op clouds vergeleken met zijn netwerk-agnostische tegenhanger. Dit werk is in submissie [107].

Om voordeel te behalen uit ongebruikte delen van het cluster en om de totale efficiëntie van het clustergebruik te verhogen, stellen wij in hoofdstuk 5 het gedistribueerde geheugen-scavenging bestandssysteem MemFSS voor. Door middel van cluster-disaggregatie is MemFSS in staat om op transparante manier beschikbaar geheugen bij compute-nodes die door andere gebruikers gereserveerd zijn bij elkaar te "sprokkelen". Hierdoor wordt de doorvoersnelheid en benutting van de computationele middelen van het totale cluster verhoogd. Bovendien is de performance-impact van geheugen-scavenging op de applicaties van andere gebruikers verwaarloosbaar. Dit werk is gepubliceerd in [113].

Samengevat presenteert dit proefschrift onze zienswijze op het optimaliseren van de executietijd van MTC applicaties vanuit het perspectief van bestandsopslag. De technieken die wij voorstellen, verbeteren in grote mate de applicatie-performance en gebruiken de beschikbare middelen op een weloverwogen manier.