Abstract

La scienza computazionale è un campo di ricerca in continua espansione. Essa combina tecnologie, metodi computazionali moderni e simulazioni per affrontare problemi troppo complessi per essere risolti efficacemente dalla sola teoria o troppo costosi o pericolosi per essere riprodotti in laboratorio. Questo settore scientifico è un campo multidisciplinare e ha un impatto su diversi problemi di scienze, ingegneria e scienze umane. Il successo dell'approccio delle scienze computazionali ha portato ad una crescente domanda di risorse di calcolo per migliorare le prestazioni delle soluzioni e permettere la crescita dei modelli, sia in termini di dimensioni che di qualità. Per queste ragioni, i paradigmi di calcolo parallelo e distribuito e lo sfruttamento del Cloud Computing sono diventati essenziali nella vita quotidiana degli scienziati computazionali. Il cloud fornisce un'enorme quantità di potenza computazionale, facilmente accessibile a tutti in modo consapevole del prezzo. Di conseguenza, la nozione di "scalabilità" di un'applicazione, che gira su sistemi paralleli o distribuiti, è diventata centrale nella scienza computazionale. In poche parole, un'applicazione è scalabile se può sfruttare in modo efficiente una quantità crescente di potenza di calcolo (ad esempio, il numero di nodi o processori). In questo dominio, una sfida di ricerca rilevante è quella di fornire scalabilità a diversi livelli, dalle librerie di software, ai framework e strumenti per aiutare la soluzione di problemi scientifici. Fornire scalabilità permette agli scienziati di affrontare problemi massicci e complessi in modo trasparente e il più semplice possibile. Questa dissertazione discute i framework e i linguaggi paralleli che permettono agli scienziati di affrontare i problemi di scienza computazionale sotto la lente del requisito di scalabilità estrema. I contributi di questo lavoro possono essere riassunti in tre categorie principali: linguaggi e strumenti, Agent-based Model (ABM) e simulazioni, e Optimization via Simulation (OvS).

Molte applicazioni scientifiche del mondo reale consistono nell'orchestrare diversi compiti indipendenti o metodi per realizzare un particolare carico di lavoro. Questi flussi di lavoro sono di solito computazionalmente e temporalmente impegnativi, quindi sfruttare tecniche parallele e distribuite è diven-

tato essenziale. Questa dissertazione presenta FLY, un linguaggio specifico per applicazioni scientifiche, che mira a riconciliare i paradigmi di Cloud e High-Performance Computing. FLY adotta un approccio multi-cloud fornendo uno strumento potente, efficace ed economico per lo sviluppo di applicazioni scientifiche scalabili basate sul flusso di lavoro. FLY sfrutta diversi e allo stesso tempo fornitori di cloud Function-as-a-Service come backend computazionali in modo trasparente. Questa dissertazione descrive il modello di programmazione di FLY, la definizione del suo linguaggio e il compilatore FLY source-to-source. Inoltre, viene discussa una valutazione delle prestazioni di FLY su un popolare benchmark per i framework di calcolo distribuito, insieme a una raccolta di casi di studio con un'analisi dei loro risultati di performance e dei costi. Infine, viene mostrato un caso d'uso reale che implementa un processo di ottimizzazione tramite simulazione utilizzando FLY per effettuare una valutazione distribuita di simulazioni su un backend AWS.

ABM è un approccio di modellazione bottom-up, dove agenti decisionali indipendenti modellano un sistema complesso. Il comportamento emergente su larga scala in ABM è influenzato dalla dimensione della popolazione e dalla complessità del comportamento di ogni agente. Tuttavia, il costo computazionale della simulazione cresce insieme all'aumento dei dettagli del modello. Questa dissertazione presenta l'architettura e l'implementazione di una libreria open-source per lo sviluppo di modelli basati su agenti utilizzando il linguaggio Rust. Rust-AB è in grado di sfruttare sia le piattaforme di calcolo sequenziale che quelle parallele. Viene discussa un'indagine sulla capacità di Rust di sviluppare simulazioni ABM e vengono descritti diversi modelli sviluppati con Rust-AB. Infine, viene anche presentato un confronto delle prestazioni contro il noto toolkit Java ABM MASON.

OvS si riferisce alle tecniche per scoprire i parametri di un modello complesso ottimizzando una o più funzioni obiettivo, che possono essere calcolate solo eseguendo una simulazione. A causa dell'alta dimensionalità dello spazio di ricerca, l'eterogeneità dei parametri e la natura stocastica della funzione di valutazione degli obiettivi, l'ottimizzazione di una simulazione di questo tipo è estremamente impegnativa dal punto di vista computazionale. Questa dissertazione discute i metodi per sfruttare la potenza di calcolo dei sistemi paralleli/distribuiti per migliorare l'efficienza e l'efficacia delle strategie OvS. In particolare, vengono presentati tre framework che differiscono per l'architettura del sistema di calcolo sottostante adottato: i) eterogeneo – dove CPU e GPU vengono utilizzate per eseguire le simulazioni in un sistema distribuito composto da un nodo eterogeneo in termini di hardware e software, ii) omogeneo – il sistema di calcolo è composto da nodi omogenei dove vengono utilizzati i software MASON (libreria di simulazione) ed ECJ (libreria

di ottimizzazione) per elaborare il processo OvS, e iii) cloud computing – il sistema di calcolo