# Progetto di High Performance Computing 2018/2019

Lorenzo Casini, matr. 0000800947
26/09/2019

## *Introduzione*

Lo scopo di questo progetto è realizzare due versioni che sfruttano la parallelizzazione sia a memoria condivisa che a memoria distribuita di un programma seriale *erathquake.c* come base di partenza.

Ho scelto di implementare la versione a memoria condivisa con OpenMP metre ho realizzato la versione a memoria distribuita con MPI.

Per comodità ho realizzato un repository github per gestire lo sviluppo del progetto sopratutto per passare in maniera agevole le modifiche sul server isi-raptor03 ed effetturare i vari test dei programmi realizzati.

Link repo: https://www.github.com/k4s0/ProgettoHPC

Per il progetto sono state usate le slide e gli esempi presentati nel corso.

## *Versione OpenMP*

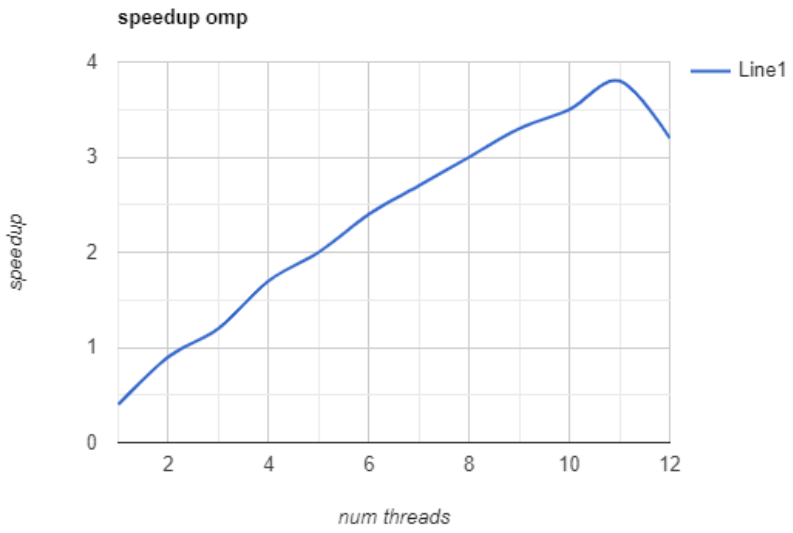
Per realizzare la versione OpenMP del programma seriale mi sono focalizzato principalmente sui vari step che quest'ultimo eseguiva in quanto ho riscontrato le maggiori criticità all'interno di esse, e se parallelizzate potevano dara il maggior beneficio in termini di performance.

(**iterazione**)

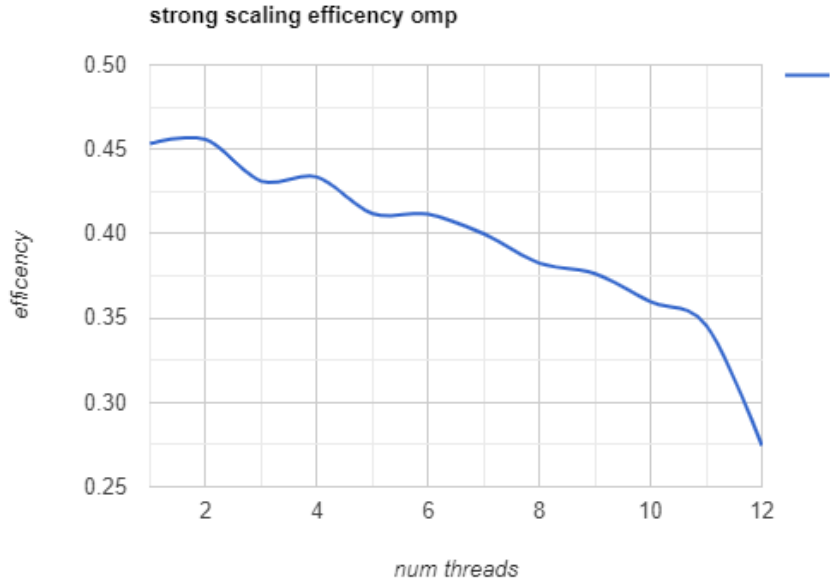
Come primo passo il programma esegue la funzione *increment_energy*, che presentava al suo interno due cicli che valorizzavano tutte le celle del dominio causando un *embarassingly parallel problem*, perché ogni singola area può essere calcolata indipendetemente dalle altre.

(**propagazione**)

Il secondo passo esegue la funzione *propagate_energy*, la prima cosa è stata l'implemetazione di una ghost area, questo ci permette di evitare un numero considerevole di controlli durante l'esecuzione. Per rompere le dipendenze fra i cicli ogni cella controlla se le celle affianco hanno soglia maggiore di EMAX. Per calcolare quante celle avessere un'energia superiore a EMAX ho utilizzato la clausola di OpenMP *reduction*.



Il grafico illustra lo speedup del programma realizzato in OpenMP, come è possibile notare lo speedup cresce all'aumentare del numero dei threads. Il grafico ci permette anche di capire in maniera evidente dove si sono presentati degli overhead, ovvero nell'utilizzo del programma con 1 solo thread e quando superiamo gli 11 thread.




Nel secondo grafico è rappresentata l'efficenza del programma in confronto al programma seriale

## *Versione MPI/CUDA*

Descrivere come il programma è stato parallelizzato sfruttando il parallelismo a memoria distribuita fornito da MPI, oppure il parallelismo massivo CUDA (solo uno dei due, a scelta). Discutere scalabilità ed efficienza della soluzione proposta.

## *Conclusioni*

Eventuali conclusioni, discussione generale dei risultati, discussione generale sul modello ecc.

## *Riferimenti bibliografici*

Nel caso in cui si sia consultata della documentazione, è utile indicarla nei riferimenti bibliografici. Ogni riferimento deve essere numerato, ed è necessario richiamarlo almeno una volta nel testo. Nel caso in cui non si sia consultata documentazione particolare, questa sezione può essere omessa.