Elaborato High Performance Computing

Giulio Golinelli
0000883007
23/02/2021

Tabella dei contenuti

1. Introduzione

Il lavoro svolto consiste nella parallelizzazione dell'algoritmo "*Skyline*" sfruttando due tecnlogie: Open-MP e MPI.
La versione seriale del programma era fornita ed è stata fortemente utilizzata come base su cui sviluppare le altre due versioni.
È stato inoltre sviluppato uno script bash chiamato "*script.sh*" per automatizzare la compilazione, esecuzione e verifica dei vari sorgenti.
Si sono eseguiti diversi test sul server *isi-raptor03.csr.unibo.it*, raccolti i rispettivi tempi di esecuzione e elaborato dei risultati su diverse caratteristiche, di seguito presentati.

N.B.: Si assume che il lettore abbia una piena conoscenza di:

- Obbiettivo e struttura dell'algoritmo skyline
- Architettura e funzionamento delle macchine multiprocessore, sia a memoria condivisa che a memoria distribuita
- Linguaggio C e fondamenti di programmazione
- Pattern di programmazione parallela
- Analisi delle prestazioni dei programmi paralleli
- Tecnologie Open-MP e MPI

N.B.: Le versioni parallelizzate degli algoritmi sono state lasciate quanto più simili alla versione seriale fornita in modo da facilitarne la comprensione e utilizzare quest'ultima come punto di riferimento nelle loro spiegazioni.

2. Dati raccolti

Tutte le tempistiche rappresentate sono state raccolte, per tutte le tecnologie, con la premura di includere la più piccola frazione di lavoro non parallelizzabile.

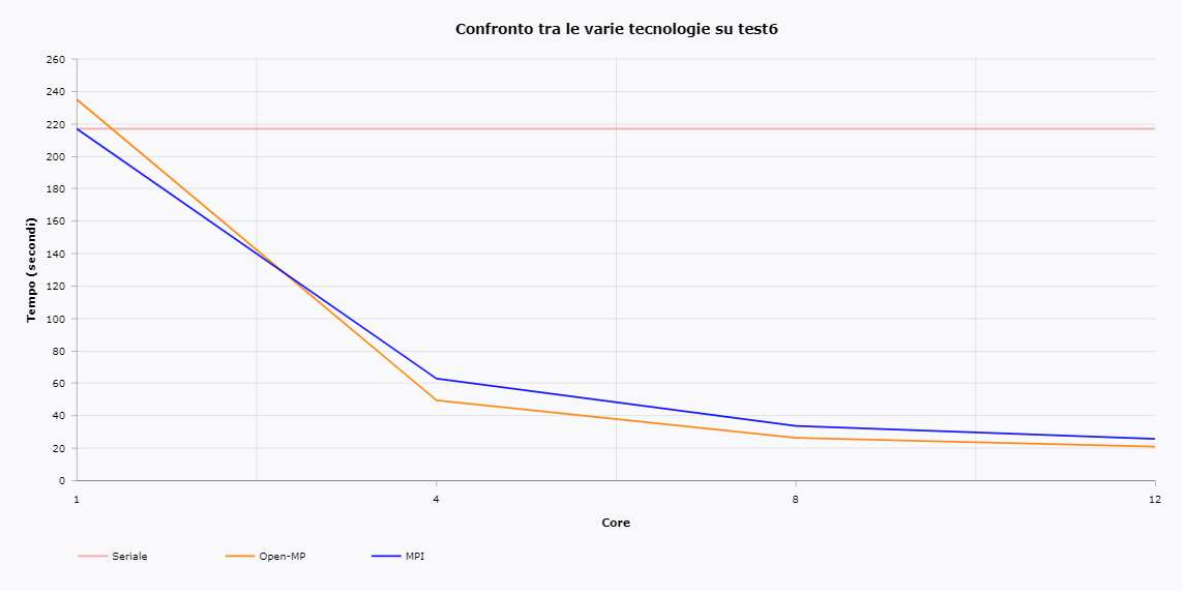

Immagine 1: Confronto tra le varie tecnologie sul datafile "test6"

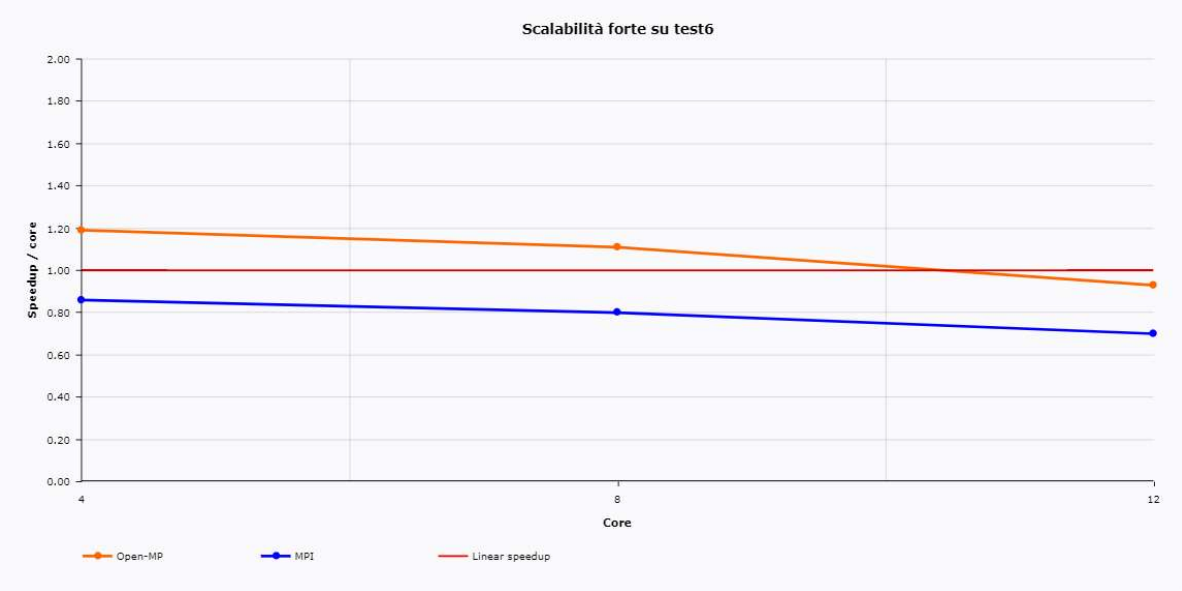

Immagine 2: Rapporto tra speedup e core sul datafile "test6"

3. versione Open-MP

3.1. Parallelizzazione

Lo speedup superlineare nella version Open-MP è probabilmente dato dal fatto che ad ogni iterazione del for esterno in cui l'if da successo, si apre un nuovo blocco Open-MP e questo crea molti costi aggiuntivi di sincronizzazione e comunicazione che con un solo processore, diventano inutili. Il for interno modifica elementi del vettore che saranno poi i soggetti delle iterazione successive del for più esterno.
Il for più esterno segue un percorso critico in quanto diventa obbligatorio che le iterazioni

del for interno avvengano prima di una sua successiva.
Diventa quindi possibile parallelizzare solamente il for più interno che comunque costituisce la maggior parte del costo computazionale dell'intero algoritmo.
Non sono state modificate le politiche di scheduling di default.
È stata inserita una operazione di riduzione per sincronizzare ogni volta il numero di punti rimanenti nello skyline.

3.2. Anlisi dei dati raccolti

Lo speedup superlineare che si osserva è dovuto al non poco onore computazionale aggiuntivo che si crea ogni qualvolta il for esterno trova un possibile candiato per lo skyline.
Ad ogni occorrenza di questa circostanza viene creata e terminata una sezione Open-MP per parallelizzare il ciclo interno e ciò accresce i costi computazionali.
Questi ultimi diventano oltretutto inutili se si utilizza la tecnologia con un solo core.
Se si calcolasse lo speedup utilizzando la versione seriale fornita (senza inutili costi aggiuntivi di Open-MP) potremmo osservare, dove è presente un speedup superlineare, un andamento invece lineare.

4. Versione MPI

4.1. Parallelizzazione

Sul sistema a memoria distribuita si è scelto di dividere lo spazio di operazione del for esterno dipendentemente dal rank.
Sia:

- `N`: Il numero di punti in input
- `size`: il numero di core utilizzati

Ogni core lavora su una porzione di `N / size` elementi dell'input opportunamente divisi, esegue il normale algoritmo e produce `N` elementi di output.
Sull'ipotesi di una corretta divisione dell'input (2 core non hanno possibilità di effettuare la medesima computazione) e relativamente ad ogni porzione di input e il rispettivo core, l'esecuzione assume un comportamento "seriale".
Il for esterno costituisce ancora un percorso critico ma ciò è assolutamente normale in una esecuzione "seriale".
Non prendendo in considerazione tutti gli elementi, l'insieme skyline di output generato da ogni core sarà un sovrainsieme dello skyline finale.
Il risultato del singolo elemento di input è espresso da un valore booleano (fa parte dello skyline o meno).
A livello implementativo è espresso sottoforma di valore intero (0 o 1) ma concettualmente non vi è differenza.
Lo skyline finale è il risultato dell'intersezione di tutti gli insiemi output di ogni core.
Viene ottenuto attraverso una riduzione che utilizza l'operatore matematico moltiplicazione tra tutti gli insiemi locali di interi (in questo caso è, sia concettualmente che in pratica, come eseguire una intersezione tra insiemi o una operazione AND tra più valori booleani).

4.2. Analisi dei dati raccolti

Gli onerosi costi di comunicazione nella tecnologia a memoria distribuita fanno si che non si verifichi mai uno speedup lineare.
Ciò è coerente anche con l'aumentare del numero di core, in quanto con questi aumenta anche la memoria che deve essere spostata (ogni core produce un output di N elementi, un core in più significa N elementi in più da condividere).

5. Conclusione

5.1. Scalabilità forte

Dai dati raccolti rappresentanti nei grafici si evince come entrambe le tecnologie si comportano coerentemente con la definizione di scalabilità forte.
Il rapporto di scalabilità, con l'aumentare di core, tende ad eguagliare il rappoorto `Lavoro del processore p / costi aggiuntivi di tecnologia` e anche questo è un sintomo assolutamente in linea con il concetto di scalabilità forte.