Relazione Progetto High Performance Computing

Shola Oshodi 0000915434

1 Introduzione

Il progetto realizzato, contiene due versioni parallele dell'algoritmo per la simulazione dell'automa cellulare Lattice Gas (HPP). La prima versione sfrutta OpenMP, mentre la seconda utilizza MPI. Osservando l'implementazione seriale del problema proposto, è possibile notare che si tratta di una computazione di tipo stencil bidimensionale a 4 punti, in cui il calcolo del passo successivo è un problema dalla struttura embarassingly parallel. Il programma quindi si dimostra particolarmente predisposto ad essere parallelizzato e come si vedrà di seguito sarà possibile ottenere discreti vantaggi in termini di tempi di esecuzione.

In Tabella 1 si illustrano le caratteristiche del server utilizzato per le analisi delle prestazioni:

Processore	Intel(R) Xeon(R) CPU E5-2603 v4
Numero di core	12
Thread per core	1
Frequenza base processore	1.70 GHz
RAM	64 GB
GPU	Nvidia GTX 1070
MPI	Versione 4.0.3
OpenMP	Versione 4.5

Table 1: Scheda tecnica server utilizzato per misurare i tempi di esecuzione

2 OpenMP

2.1 Strategie di parallelizzazione

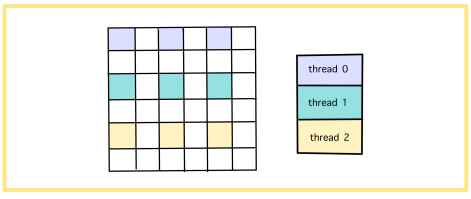

Figure 2.1: Ripartizione della matrice tra i diversi threads

Nella versione a memoria condivisa si è deciso di sfruttare la struttura embaraslingly parallel del calcolo del passo per suddividere equamente lo scorrimento della matrice tra i diversi threads (Figura 2.1) utilizzando la clausola omp parallel for. I due cicli sono stati uniti in uno solo utilizzando collapse, si è scelto inoltre di utilizzare un'assegnazione statica in quanto il numero di passi da calcolare era già noto in fase di compilazione. Per la chunk size si è ritenuto appropriato mantenere il valore predefinito.

2.2 Analisi delle prestazioni

Le misurazioni presenti in Tabella 2 utilizzate per il calcolo di Speed up e Strong Scaling, sono state ottenute impostando un valore di input N=1024 e S=256,il valore utilizzato per il calcolo è dato dalla media di 5 misurazioni. Come è possibile notare in Figura 2.3 lo speed up mantiene un andamento quasi lineare, con una crescita che inizia a rallentare a partire da p=7. Dopo aver raggiunto il picco

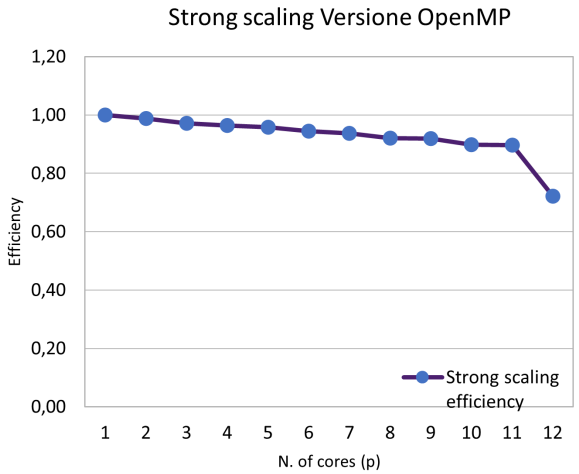

Figure 2.2: Strong Scaling - OpenMp

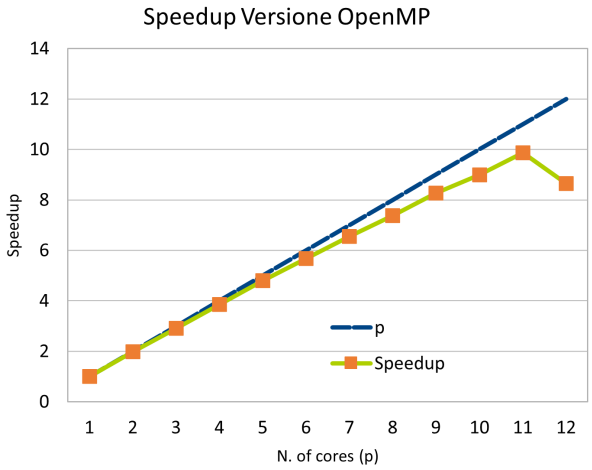

Figure 2.3: Speed up - OpenMp

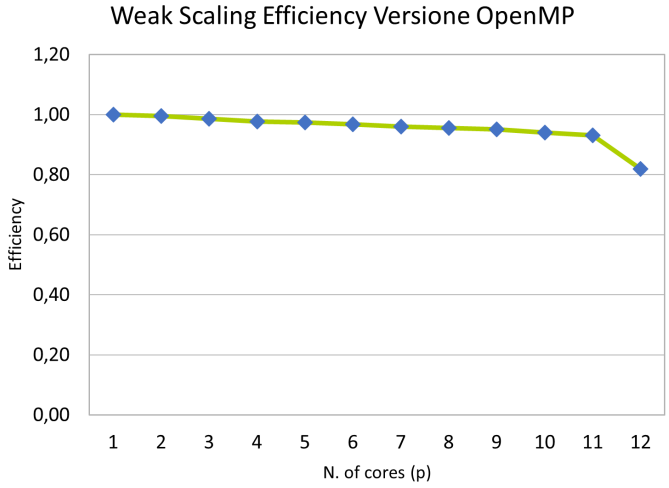

Figure 2.4: Weak Scaling Efficiency - OpenMp

p	t1	t2	t3	t4	t5	Average	Speed up	Strong Scaling Efficiency
1	6,03	6,07	6,03	6,07	6,08	6,06	1,00	1,00
2	3,07	3,06	3,07	3,05	3,08	3,07	1,98	0,99
3	2,08	2,06	2,10	2,08	2,07	2,08	2,91	0,97
4	1,57	1,58	1,56	1,57	1,58	1,57	3,85	0,96
5	1,27	1,26	1,26	1,27	1,26	1,26	4,79	0,96
6	1,07	1,06	1,08	1,07	1,06	1,07	5,67	0,95
7	0,92	0,93	0,93	0,92	0,92	0,92	6,55	0,94
8	0,82	0,82	0,83	0,81	0,83	0,82	7,37	0,92
9	0,73	0,74	0,72	0,73	0,74	0,73	8,27	0,92
10	0,67	0,69	0,66	0,67	0,68	0,67	8,99	0,90
11	0,62	0,61	0,61	0,62	0,61	0,61	9,86	0,90
12	0,57	0,59	0,86	0,59	0,89	0,70	8,65	0,72

Table 2: Misurazioni Speed up e Strog Scaling - OpenMp

p	t1	t2	t3	t4	t5	Average	Weak Scaling Efficiency
1	6,04	6,04	6,03	6,04	6,03	6,04	1,00
2	6,07	6,06	6,06	6,07	6,07	6,07	1,00
3	6,12	6,12	6,12	6,13	6,12	6,12	0,99
4	6,17	6,17	6,18	6,18	6,18	6,19	0,98
5	6,20	6,21	6,21	6,20	6,20	6,20	0,97
6	6,24	6,25	6,24	6,23	6,24	6,24	0,97
7	6,28	6,28	6,30	6,28	6,29	6,29	0,96
8	6,30	6,32	6,33	6,32	6,31	6,32	0,96
9	6,32	6,38	6,35	6,36	6,36	6,35	0,95
10	6,41	6,44	6,41	6,44	6,40	6,42	0,94
11	6,51	6,47	6,47	6,48	6,50	6,49	0,93
12	7,40	7,41	7,29	7,32	7,46	7,38	0,82

Table 3: Misurazioni Weak scaling - OpenMp

con p=11, si può osservare un calo consistente dello speed up quando p=12, questo peggioramento si presenta perché,per l'input preso in esame, a partire da p=12 si ha un'incidenza maggiore dell'overhead che influisce negativamente sulle prestazioni. Si può osservare che ciò si riflette anche sull'efficienza della Strong Scaling e della Weak Scaling che mantengono un andamento praticamente lineare fino all'ultima esecuzione (Figura 2.2 e Figura 2.4). Per la misurazione della Weak scaling si è scelto N0=1024 e s=256.

3 MPI

3.1 Strategie di parallelizzazione

Nella versione a memoria distribuita, si è cercato di dividere il dominio in maniera uniforme tra i processi MPI, e ad ogni processo è stato assegnato l'aggiornamento del vicinato pari e vicinato dispari della porzione di dominio di sua competenza. Per partizionare la matrice di input si è utilizzata ScatterV, in modo da poter assegnare al processo di rango 0 un numero maggiore di righe nel caso in cui la dimensione della griglia N non fosse multipla del numero di processi. Essendo il calcolo della fase dispari dipendente dal risultato restituito dalla fase pari, e avendo scelto di partizionare la matrice di input per righe, è stato necessario aggiungere ghost cells per la comunicazione tra processi adiacenti. Per questo motivo sono state utilizzate due Sendrecv, la prima per trasferire i valori necessari nelle ghost cells dal processo precedente a quello successivo e la seconda dal processo successivo al precedente per per aggiornare il contenuto dell'ultima riga della matrice locale di ogni processo. Si è poi utilizzata GatherV per unire tra loro le matrici locali in modo che fosse possibile procedere alla scrittura dell'immagine PGM. Non è stato necessario implementare nuovamente Scatterv ad ogni passo in quanto ogni processo manteneva ancora in memoria i dati locali necessari per le computazioni successive. In Figura 3.1 si riassume il procedimento

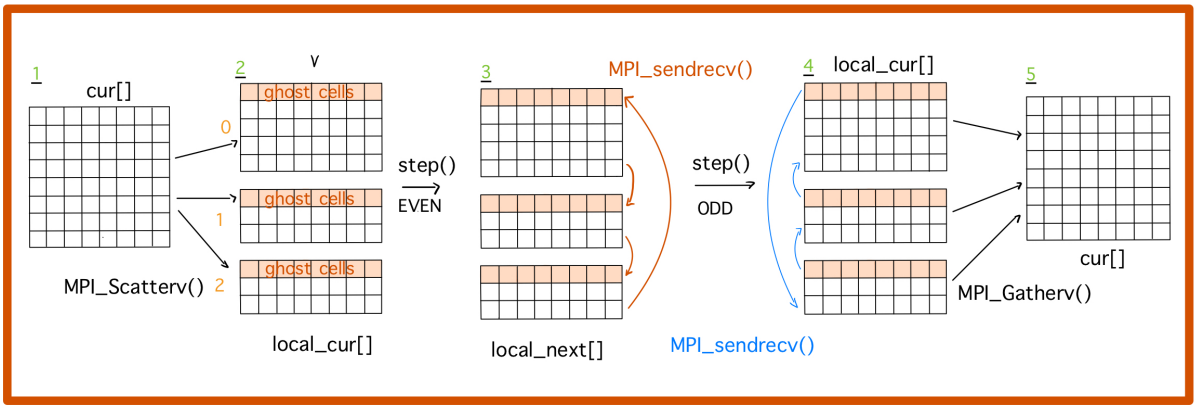

Figure 3.1: Procedimento parallelizzazione MPI

applicato.

3.2 Analisi delle prestazioni

Anche in questo caso i valori ottenuti in Tabella 4 e in Tabella 5 sono stati calcolati utilizzando la media di 5 esecuzioni impostando un input N=1024 e s=256 per Speed up e Strong Scaling Efficiency e N0=1024 s=256 per Weak Scaling Efficiency. Il programma presenta una buona efficienza complessiva e uno speed up quasi lineare (Figura 3.2)fino a p=8. In questo caso il motivo per cui si ha un rallentamento nella crescita delle prestazioni all'aumentare di p è a causa delle diverse comunicazioni che sono necessarie ai processi per sincronizzarsi e per scambiare i risultati delle computazioni. L'andamento della retta della weak scaling efficiency, che non si discosta troppo da quello ideale fino a p=7 circa, mostra che il programma è in grado di distribuire abbastanza bene le computazioni tra i diversi processori raggiungendo un valore minimo di 0,72 per p=12. Il picco dello speed up raggiunge a p=12 con un valore di 8,72.

4 Conclusioni

In questo progetto sono state realizzate due versioni parallele dell'implementazione dell'automa cellulare di Lattice Gas, entrambe le versioni hanno mostrato sensibili miglioramenti delle prestazioni rispetto alla versione seriale. E' stato inoltre possibile osservare con chiarezza l'impatto che overhead e comunicazioni fra processi possono avere sui tempi di esecuzione, per questo motivo per sfruttare a pieno i vantaggi della parallelizzazione è importante bilanciare adeguatamente numero di processi rispetto alla dimensione del problema.

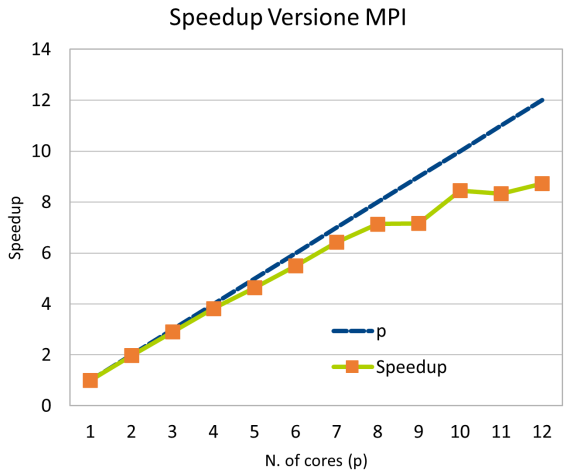

Figure 3.2: Speed up - MPI

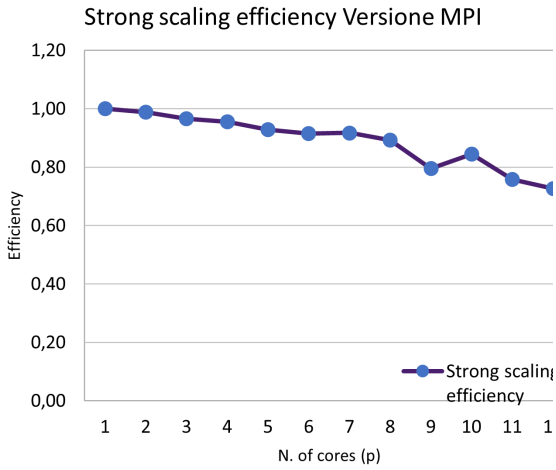

Figure 3.3: Strong Scaling - MPI

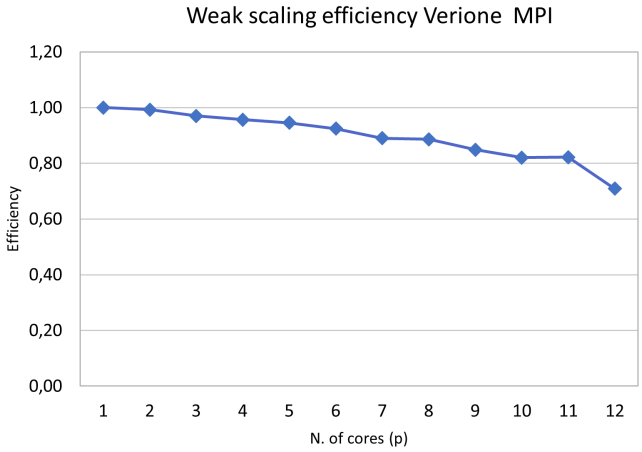

Figure 3.4: Weak Scaling Efficiency - MPI

p	t1	t2	t3	t4	t5	Average	Speed up	Strong Scaling Efficiency
1	4,83	5,02	4,83	4,82	4,83	4,87	1,00	1,00
2	2,44	2,45	2,53	2,44	2,45	2,46	1,98	0,99
3	1,68	1,68	1,67	1,68	1,69	1,68	2,90	0,97
4	1,28	1,27	1,28	1,26	1,28	1,27	3,82	0,95
5	1,04	1,05	1,04	1,05	1,06	1,05	4,64	0,93
6	0,89	0,89	0,88	0,88	0,89	0,89	5,49	0,92
7	0,78	0,77	0,69	0,78	0,77	0,76	6,42	0,92
8	0,68	0,68	0,69	0,68	0,68	0,68	7,13	0,89
9	0,68	0,66	0,68	0,67	0,71	0,68	7,16	0,80
10	0,57	0,58	0,57	0,58	0,58	0,58	8,45	0,84
11	0,62	0,57	0,56	0,57	0,60	0,58	8,33	0,76
12	0,55	0,58	0,56	0,55	0,55	0,56	8,72	0,73

Table 4: Misurazioni Speed up e strong scaling - MPI

p	t1	t2	t3	t4	t5	Average	Weak Scaling Efficiency
1	4,82	4,83	4,83	4,83	4,82	4,83	1,00
2	4,86	4,86	4,87	4,87	4,86	4,86	0,99
3	4,98	4,98	4,97	4,98	4,97	4,98	0,97
4	5,04	5,05	5,04	5,05	5,05	5,05	0,96
5	5,10	5,11	5,11	5,10	5,11	5,11	0,95
6	5,21	5,22	5,22	5,23	5,22	5,22	0,92
7	5,41	5,43	5,44	5,46	5,42	5,44	0,89
8	5,45	5,45	5,44	5,46	5,42	5,43	0,89
9	5,69	5,70	5,69	5,68	5,68	5,69	0,85
10	5,91	5,87	5,88	5,90	5,83	5,87	0,82
11	5,84	5,90	5,89	5,90	5,83	5,87	0,82
12	6,63	7,03	6,99	6,63	6,76	6,81	0,71

Table 5: Misurazioni Weak scaling - MPI