Lattice QCD e GPU

M. D'Elia (Pisa)

Qualche anno fa, stimolati dalla necessità di calcolo dopo la chiusura di apeNEXT, abbiamo esplorato, come altri gruppi nel mondo, la possibilità di studiare LQCD sulle GPU. (C. Bonati, G. Cossu, M.D., A. Di Giacomo)

Il codice, sviluppato in CUDA per la singola GPU, ci ha permesso di studiare con successo una serie di questioni riguardanti la QCD a temperatura e densità finita, anche in presenza di campi esterni.

C. Bonati, G. Cossu, M. D'Elia, P. Incardona, Comput. Phys. Commun. 183, 853 (2012)

C. Bonati, G. Cossu, M. D. and F. Sanfilippo, Phys. Rev. D 83, 054505 (2011).

C. Bonati, P. de Forcrand, M. D., O. Philipsen and F. Sanfilippo, arXiv:1201.2769 [hep-lat]

M. D., M. Mariti and F. Negro, Phys. Rev. Lett. 110:082002 (2013)

C. Bonati, M. D., M. Mariti, F. Negro, F. Sanfilippo, Phys. Rev. Lett. 111:182001 (2013)

GPU hardware: GPU farm a Pisa, Genova e QUONG cluster a Roma

Caratteristiche principali:

- **Quasi tutto il codice gira sulla GPU, riducendo al minimo i trasferimenti di memoria GPU-CPU**
- **All'interno della GPU, il calcolo richiede di trasferire dati dalla global memory ai core, la capacità di trasferimento può limitare la capacità di calcolo.**
- **Per ottimizzare le potenzialità della GPU, il codice usa la singola precisione a parte dove è necessaria la doppia (test di Metropolis alla fine della dinamica molecolare).**
- **Il nostro codice è scritto per fermioni staggered. Il costo maggiore è l'applicazione dell'operatore di Dirac, che richiede circa di trasferire un byte per ogni operazione floating point. La performance del nostro codice è quindi limitata dalla banda passante, e scala circa come questa nelle evoluzioni tecnologiche delle GPU. Attualmente siamo intorno ai 200 GB/s $\longrightarrow$ 200 Gflops**

Domande del giorno:

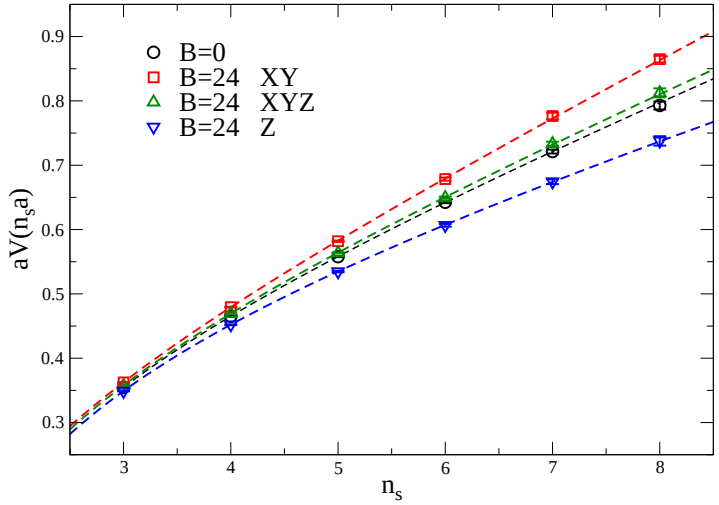
- Abbiamo necessità di passare ad una implementazione multi-GPU?

- La necessità di far comunicare più spesso le GPU con il mondo esterno (in particolare fra di loro) sarà fatale oppure no?

Infatti i link di comunicazione con l'esterno sono tipicamente più di un ordine di grandezza più lenti del trasferimento di memoria interno (qualche GB/s)

Il modo migliore di impostare la discussione è di considerare un caso fisico di interesse per l'immediato futuro.

Il caso di interesse



Recentemente abbiamo studiato le modifiche indotte da un campo magnetico esterno sul potenziale statico quark-antiquark, che diventa anisotropo (C. Bonati, M. D., M. Mariti, M. Mesiti, F. Negro, F. Sanfilippo, arXiv:1403.6094).

Lo studio è stato condotto con reticoli fino a 40^4, lattice spacing $a = 0.125$ fm, $2+1$ flavors con masse fisiche dei quark (discretizzazione con stout smearing), sul BG/Q

In futuro vorremmo dare seguito a questo studio esplorativo su reticoli più fini, almeno $48^3 \times 96$ a lattice spacing $a = 0.1$ fm

il BG/Q è la risorsa ideale per un tale calcolo oggi. Ma fra due anni, cosa avremo al posto del BG/Q?

Per questioni di memoria, un tale sistema non può essere allocato su una singola GPU:

- **La sola configurazione di gauge ($48^3 \times 96$) prende 5-6 GB, tenendo conto dei livelli di stout-smearing e dei vari campi pseudofermionici si arriva ad un totale di 25-30 GB**
- **La GPU K20 ha una capienza massima di 5-6 GB. La K40 12 GB**

Per capire come allocarlo al meglio su più GPU, con la speranza di non rimetterci troppo in termini di comunicazioni, va fatto uno studio preliminare sui tempi di comunicazione confrontati con i tempi di calcolo

(C. Bonati, E. Calore, A. Carboni, S. Coscetti, M.D., M. Mariti, M. Mesiti, F. Negro, F. Schifano, F. Sanfilippo, R. Tripiccione)

Solo se $t_{comunicazione} \lesssim t_{calcolo}$ si può sperare di poter scrivere un algoritmo furbo che mascheri le comunicazioni nei tempi di calcolo.

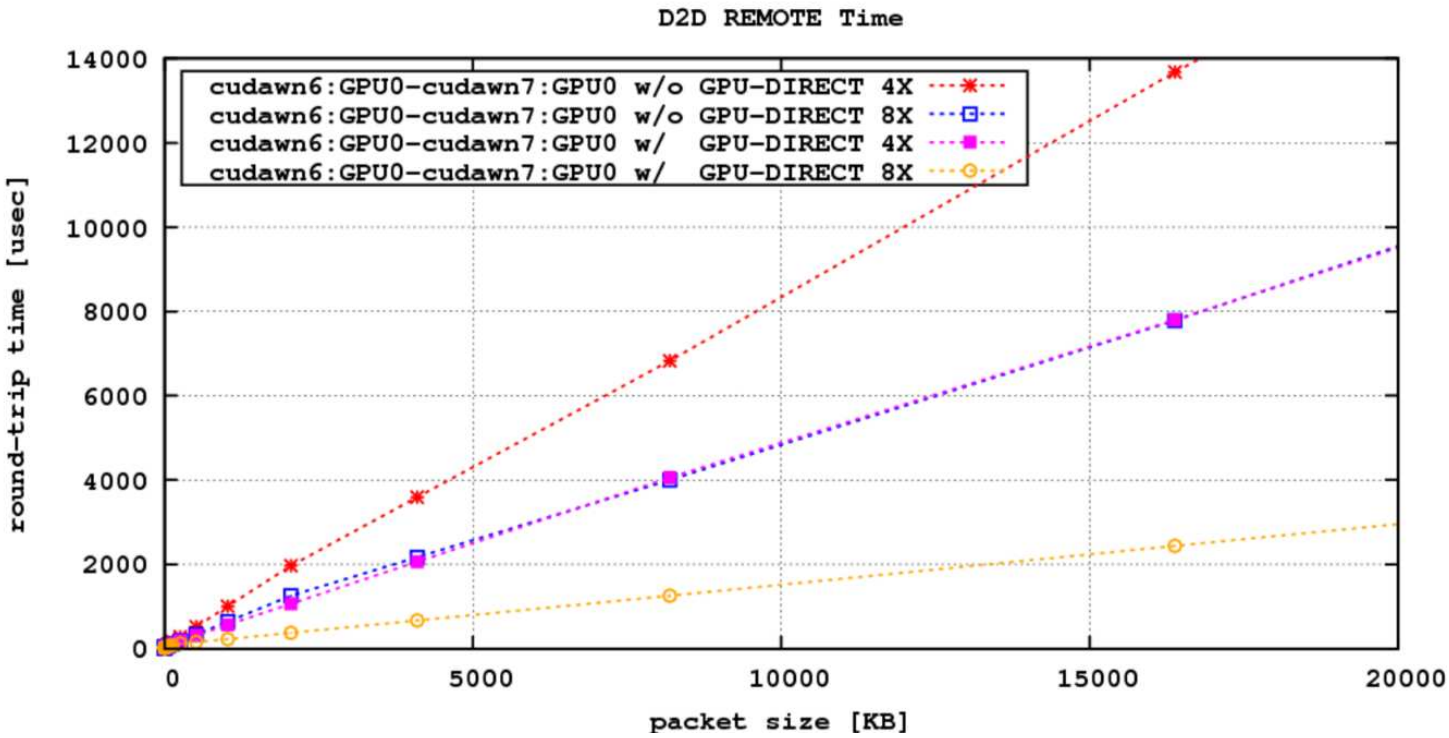
L'oggetto elementare più importante da analizzare è l'applicazione dell'operatore di Dirac (staggered) ad un campo fermionico.

Consideriamo il caso in singola precisione in quanto è il caso più rilevante nell'implementazione GPU. Supponiamo di aver diviso il reticolo globale in reticoli locali di taglia L^4.

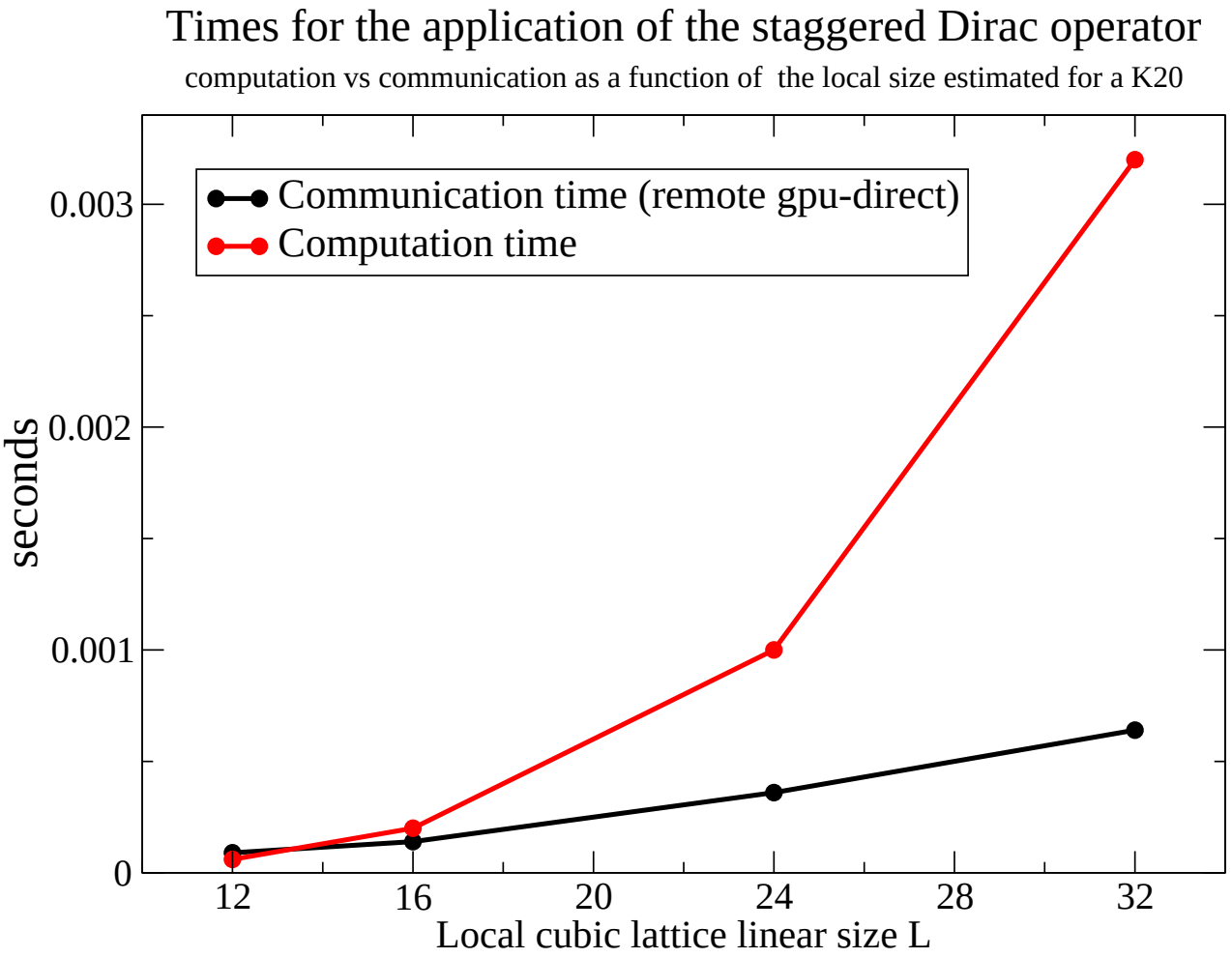
- **Bisogna, per ogni sito, moltiplicare 8 (una avanti e una indietro per ogni direzione) matrici 3×3 complesse per vettori complessi a 3 componenti presi dai siti primi vicini:**
  $L^4\ 584$ operazioni floating point.
  Ad esempio, per $L = 24$ con una performance di 200 Gflops per GPU, questo richiede 10^{-3} secondi.

- **L'operatore viene applicato ripetutamente nell'inversione della matrice di Dirac: le matrici 3×3 sono sempre le stesse, i vettori cambiano e vanno aggiornati**
  Ad ogni applicazione ogni GPU deve recuperare i vettori da 8 slice di L^3 siti dai sottoreticoli primi vicini (8 pacchetti di $24L^3$ byte di comunicazione)

Per i tempi di comunicazione, consideriamo il caso (meno favorevole) di GPU poste su nodi diversi. L'uso di comunicazioni GPU-DIRECT riduce i tempi notevolmente

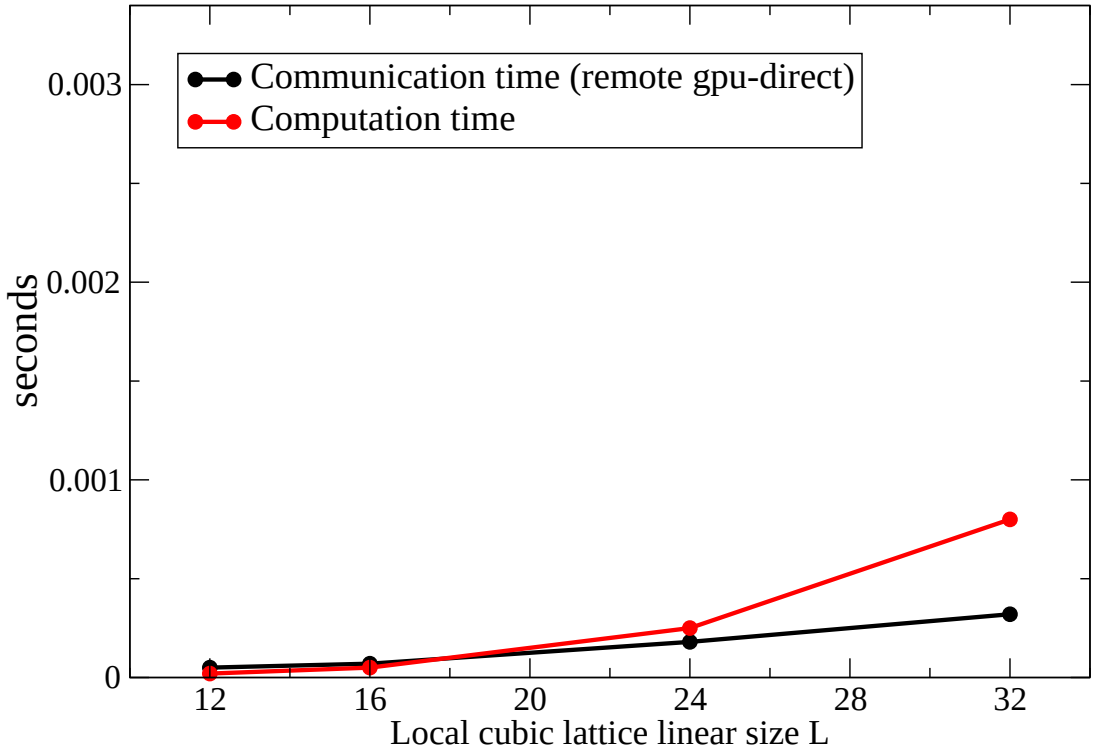


Alcuni test eseguiti da Fabio Schifano



Un calcolo grossolano mostra che già per reticoli locali 24^4 (corrispondenti a dividere il $48^3 \times 96$ su 32 GPU) i tempi di calcolo permettono ampiamente di mascherare i tempi di comunicazione. Ma si possono esplorare anche soluzioni più distribuite (O(100) GPU)

Chiaramente dobbiamo tener conto che nei prossimi due anni la tecnologia potrebbe cambiare considerevolmente, la potenza di calcolo potrebbe crescere più velocemente della velocità di trasferimento



ipotesi (Pascal, 2016): banda passante della memoria interna a 1TB/s $\longrightarrow$ 1 Tflops sostenuto per GPU, più aumento di un fattore 2 nelle comunicazioni (ad esempio PCI-express2 $\longrightarrow$ PCI-express3 ?)

Reticoli locali dell'ordine di 24^4 sembrano ancora ragionevoli. Ma potrebbero esserci anche grossi benefit (ad esempio memoria CPU-GPU condivisa in Pascal?)

Conclusioni

- E' ragionevole mettersi al lavoro per parallelizzare in modo efficiente un reticolo di lato circa 5 fm ed uno spacing $a \leq 0.1$ fm su 32 o più GPU

- Chiaramente, una volta scritto in modo efficiente l'operatore di Dirac, altre parti, non discusse qui, potrebbero diventare importanti (ad esempio calcolo delle staples per la parte di pura gauge) ma non essenziali.

- Siamo al lavoro alla scrittura del codice vero e proprio ...