Università di Bologna, Laurea in Ingegneria e Scienze Informatiche

Corso di High Performance Computing

Progetto d'esame 2023/2024

prof. Moreno Marzolla

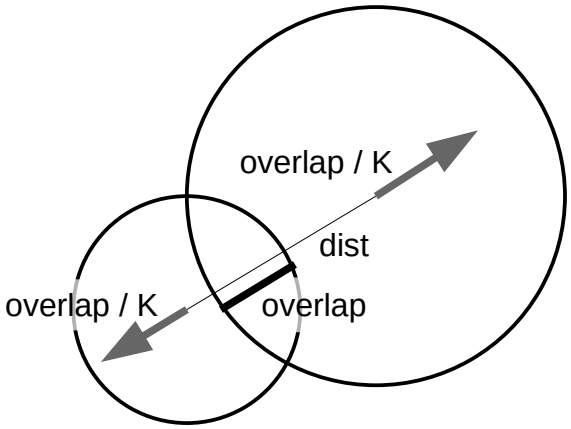
Versione 1.0 del 13/12/2023

Prima versione

1. Intersezione di cerchi

Viene fornito un programma seriale che genera N cerchi di posizione e raggio casuali, distribuiti su una regione del piano; i cerchi possono sovrapporsi, in tutto o in parte. Scopo del programma è spostare i cerchi quel tanto che basta affinché non si sovrappongano; più precisamente, vogliamo determinare una nuova collocazione dei cerchi che eviti sovrapposizioni e minimizzi l'area del poligono che li contiene. La versione 3D di questo algoritmo ha numerose applicazioni per il calcolo del minimo volume occupato da un mezzo granuloso incomprimibile (sabbia, cereali, combustibile solito per razzi...).

La soluzione proposta si basa su un algoritmo di tipo "force-directed" [1]: due cerchi che si sovrappongono esercitano una forza repulsiva l'uno verso l'altro; il programma determina la somma vettoriale di tutte le forze che agiscono su ciascun cerchio, e successivamente lo sposta lungo la risultante. Il procedimento viene iterato per un numero di passi definito dall'utente; tuttavia, l'algoritmo non trova necessariamente la soluzione ottima.



La parte dell'algoritmo computazionalmente più onerosa è la funzione `compute_forces()` di cui riportiamo lo pseudocodice (K è una costante definita nel programma fornito):

```
compute_forces( ): integer
      integer n_intersections := 0;
      for i := 0 to N-1 do
            for j := i + 1 to N-1 do
```

```
            if (i cerchi c[i] e c[j] si sovrappongono) then
                  n_intersections := n_intersections + 1;
                  overlap := lunghezza sovrapposizione tra c[i] e c[j];
                  c[i].dxy := c[i].dxy - overlap/K;
                  c[j].dxy := c[j].dxy + overlap/K;
            endif
      endfor
endfor
return n_intersections;
```

La funzione confronta tutte le coppie (c_i, c_j) con $i < j$; per ogni coppia di cerchi che si sovrappongono, si calcola la lunghezza della porzione sovrapposta, e aggiunge una frazione di tale lunghezza a ciascuno dei due con direzioni opposte (si notino i segni). La funzione restituisce il numero di coppie che si sovrappongono. Al termine di `compute_forces()` viene eseguita una diversa procedura che modifica la posizione dei cerchi in base alla risultante degli spostamenti.

Esistono soluzioni più efficienti per questo problema, ma ai fini del progetto si richiede di attenersi alla versione fornita, che ha costo quadratico in N ma è relativamente facile da parallelizzare.

Il programma seriale può essere eseguito come:

```
./circles [ncircles [iterations]]
```

dove *ncircles* è il numero di cerchi da generare, e *iterations* il numero di iterazioni da eseguire. Il programma è in grado di salvare la posizione di ogni cerchio in un file che successivamente può essere elaborato con gnuplot per produrre una rappresentazione grafica che è possibile usare per produrre un filmato. Ai fini del progetto non bisogna generare i file di dati, perché ciò costituirebbe una fase di I/O molto pesante che influenza negativamente le prestazioni delle versioni parallele.

Si faccia riferimento ai sorgenti per le istruzioni di compilazione.

2. Cosa consegnare

Viene richiesto di consegnare due versioni parallele del programma seriale fornito:

1. La prima, chiamata `omp-circles.c`, deve utilizzare OpenMP;
2. La seconda deve utilizzare, a scelta, MPI oppure CUDA (uno dei due, non entrambi). Il file deve chiamarsi rispettivamente `mpi-circles` oppure `cuda-circles`

Oltre ai due programmi di cui sopra, è obbligatorio includere:

3. Una relazione in formato PDF, della **lunghezza massima di sei facciate,** che descriva le strategie di parallelizzazione adottate e discuta le prestazioni dei programmi realizzati usando le metriche più appropriate.

Ulteriori requisiti:

- I sorgenti devono essere adeguatamente commentati. All'inizio di ogni file deve essere indicato cognome, nome e numero di matricola dell'autore/autrice.
- Includere un file di testo chiamato README contenente le istruzioni per la compilazione e l'esecuzione dei programmi consegnati; chi lo desidera può usare un makefile per la compilazione (non obbligatorio).
- Includere ogni altro file necessario alla compilazione del software (es., `hpc.h` per chi ne fa uso). È possibile scomporre il proprio codice in più sorgenti da compilare separatamente e

riunire in fase di linking; in tal caso occorre documentare la procedura di compilazione nel file README di cui al punto precedente.

- I programmi verranno compilati e testati sul server `isi-raptor03.csr.unibo.it`. Tuttavia, le misure di prestazioni descritte nella relazione (vedi oltre) non devono necessariamente essere condotte sul server, ma possono essere effettuate su proprio hardware.
- Tutti i programmi devono compilare senza *warning*. Per le versioni OpenMP e MPI si usino i flag `-std=c99 -Wall -Wpedantic`, come fatto durante il corso.
- Verranno accettati programmi che impongono vincoli sulla dimensione del dominio (es., numero di cerchi multiplo di…), tenendo presente che comporteranno una valutazione inferiore. La relazione deve indicare chiaramente l'esistenza di tali vincoli.
- La relazione non deve superare la lunghezza di **sei facciate in formato A4**, contando tutte le pagine (inclusi eventuali frontespizi, indici o altro; suggerisco di evitare frontespizi e indici, che su un documento così corto non servono). Il formato della relazione è libero, ma viene fornito uno schema in formato LibreOffice che può essere usato come base se lo si desidera.
- La relazione non deve descrivere il codice riga per riga (per quello ci sono già i sorgenti), ma le strategie di parallelizzazione adottate, eventualmente con l'ausilio di schemi o diagrammi. La relazione **deve** riportare e commentare le prestazioni delle versioni parallele realizzate utilizzando le metriche che si ritengono più appropriate. Indicare il proprio cognome, nome e numero di matricola.

Invito a prestare la massima attenzione alla qualità della relazione, perché una relazione scadente èil motivo più frequente di penalizzazione della valutazione. Vanno evitati (e saranno fortemente penalizzati) errori grossolani come “~~ghost shell~~” invece di “ghost cell”, “~~pull di thread~~” invece di “pool di thread”. Va inoltre evitato l'uso dei seguenti termini:

- “tempistica/tempistiche” (→ tempi di esecuzione)
- “prestante” o “performante” (“~~algoritmo più prestante~~” / “~~algoritmo più performante~~”→ algoritmo con prestazioni migliori)
- “randomico” (→ casuale)

Anche se la relazione del progetto non è una tesi di laurea, è utile prendere visione dei consigli per la stesura della tesi, limitatamente alla parte relativa ai consigli di scrittura. Può anche essere utile fare riferimento alla guida di stile di programmazione che viene adottata nel laboratorio di Algoritmi e Strutture Dati.

3. Modalità di svolgimento del progetto

- Il progetto deve essere frutto del lavoro individuale di chi consegna. Non è consentito condividere il codice o la relazione con altri, in tutto o in parte.
- Il programma fornito può essere modificato a piacimento; non bisogna però snaturare l'algoritmo, che deve avere costo quadratico in N per ogni iterazione.
- È ammesso l'uso di porzioni di codice trovato in rete o da altre fonti purché (*i*) la provenienza di codice scritto da terzi sia chiaramente indicata in un commento, e (*ii*) la licenza di tale codice ne consenta il riutilizzo. L’unica eccezione è il codice fornito dal docente a lezione o in laboratorio, che può essere usato liberamente senza necessità di indicarne la fonte.
- Le richieste di chiarimenti sulle specifiche (cioè su questo documento) vanno inviate sul forum che è stato creato sulla piattaforma “Virtuale”; richieste inviate via mail non riceveranno risposta. Si tenga presente che non sarà possibile rispondere a domande relative al proprio codice, né a richieste di debug: il progetto è una componente essenziale dell'esame, e deve essere svolto in totale autonomia.

4. Consegna e validità del progetto

Il progetto può essere consegnato in qualsiasi momento, prima o dopo aver sostenuto la prova scritta. Le valutazioni positive (del progetto e della prova scritta) restano valide **fino al 30 settembre 2024**; dopo tale data inizierà la nuova edizione del corso e tutti i voti in sospeso saranno annullati.

Il progetto si consegna una volta sola. Non è possibile apportare modifiche o correzioni dopo la consegna: chi vuole migliorare la valutazione dovrà consegnare un nuovo progetto su nuove specifiche assegnate dal docente.

La consegna deve avvenire tramite la piattaforma https://virtuale.unibo.it/, caricando un unico archivio in formato .zip oppure .tar.gz contenente sia i sorgenti che la relazione. L'archivio dovrà essere denominato con il cognome e nome dell'autore/autrice (es., `MarzollaMoreno.zip` oppure `MarzollaMoreno.tar.gz`), e dovrà contenere una directory con lo stesso nome (es., `MarzollaMoreno/`) contenente a sua volta i file secondo la seguente struttura:

`MarzollaMoreno/src/hpc.h` (*se usato*)

`MarzollaMoreno/src/omp-circles.c`

`MarzollaMoreno/src/mpi-circles.c` (*se si svolge la versione MPI*)

`MarzollaMoreno/src/cuda-circles.cu` (*se si svolge la versione CUDA*)

`MarzollaMoreno/src/Makefile` (*facoltativo*)

`MarzollaMoreno/README`

`MarzollaMoreno/Relazione.pdf`

Includere ogni altro file necessario alla compilazione, inclusi altri sorgenti nel caso in cui si ricorra a compilazioni separate.

La piattaforma Virtuale limita a 20 MB la dimensione dei file caricati, che dovrebbe comunque essere ampiamente sufficiente.

5. Valutazione dei progetti

Un progetto verrà considerato **sufficiente** se soddisfa almeno i seguenti requisiti minimi:

- I programmi compilano ed eseguono correttamente su istanze di input anche soggette a vincoli (es., dimensione del dominio multipla di...) sul server `isi-raptor03.csr.unibo.it`.
- La relazione dimostra un livello sufficiente di padronanza degli argomenti trattati ed è scritta in modo adeguato (chiarezza, correttezza grammaticale, uso appropriato della punteggiatura, uso corretto della lingua). Chi non è madrelingua italiana può scrivere la relazione in inglese se lo ritiene utile.

Ulteriori aspetti che potranno comportare una valutazione superiore:

- Qualità del codice, in termini di generalità, chiarezza, ed efficienza (es., uso appropriato delle primitive e/o dei pattern di programmazione parallela appropriati ed efficienti).
- Qualità della relazione, in termini di correttezza, chiarezza, completezza e presentazione, ovviamente tenuto conto della lunghezza massima del documento.

Ai fini di una valutazione elevata non è indispensabile che i programmi consegnati siano i più efficienti possibile; verrà piuttosto valutata la capacità di utilizzare i pattern di programmazione parallela più adeguati al problema e al paradigma di programmazione adottato, la chiarezza della relazione, nonché la correttezza delle metriche adottate per la misura delle prestazioni.

Chi desidera approfondire e applicare tecniche che non sono state viste a lezione è benvenuto/a, ma occorre osservare che questo da solo non garantisce una valutazione migliore, soprattutto se sono

presenti errori o si dimostrano carenze nelle competenze di base (es., l'analisi delle prestazioni non è corretta, il codice presenta dei bug, ecc.). Di conseguenza, prima di fare cose extra è importante assicurarsi che la parte obbligatoria del progetto sia stata svolta correttamente.

Sebbene il progetto possa essere consegnato in qualsiasi momento, effettuerò tre sessioni di valutazione al termine delle sessioni d'esame di gennaio/febbraio 2024, giugno/luglio 2024 e settembre 2024. Questo significa che alla fine di febbraio 2024, luglio 2024 e settembre 2024 valuterò tutti i progetti ricevuti fino a quel momento. Chi avesse delle scadenze, ad esempio per avere borse di studio o per laurearsi, è pregato/a di segnalarmelo alla consegna. È tuttavia obbligatorio consegnare il progetto almeno **10 giorni lavorativi prima della data entro la quale si richiede la correzione** (sottolineo **lavorativi)** per consentirmi di far fronte ad eventuali altri impegni accademici.

Non sarò disponibile durante il mese di agosto 2024, per cui si prega di tenere conto che in tale periodo non sarò in grado di valutare progetti né verbalizzare voti.

6. Checklist per la consegna

Prima di consegnare il progetto, assicurarsi che i requisiti fondamentali elencati nella lista seguente siano soddisfatti:

- ☐ Vengono consegnate due versioni del programma, una che usa OpenMP, e una che usa (a scelta) MPI oppure CUDA.
- ☐ I sorgenti compilano ed eseguono correttamente sul server `isi-raptor03.csr.unibo.it`.
- ☐ La relazione è in formato PDF e ha lunghezza minore o uguale a 6 facciate.
- ☐ I sorgenti e la relazione indicano chiaramente cognome, nome e numero di matricola dell'autore/autrice.
- ☐ Il progetto viene consegnato in un unico archivio .zip oppure .tar.gz, nominato con nome e cognome dell'autore, che include i sorgenti e la relazione in formato PDF.

7. Riferimenti bibliografici

[1] *Force-Directed Graph Drawing* https://en.wikipedia.org/wiki/Force-directed_graph_drawing