Esame di Programmazione su Architetture Parallele

Metodo del simplesso per la risoluzione della programmazione lineare

Belliato Riccardo (mat. 142652) Simone Tomada

2022-08-03

Abstract

In questa relazione si propone una implementazione del metodo del simplesso a due fasi in CUDA per la risoluzione dei problemi di programmazione lineare in forma canonica.

Dopo una breve descrizione dell'algoritmo, seguirà la discussione su alcune scelte implementative.

Infine verranno valutate performance e scalabilità della soluzione proposta confrontando i tempi di esecuzione dell'algoritmo su istanze a dimensione crescente generate casualmente.

Contents

Introduzione al metodo del simplesso

Problemi di ottimizzazione lineare

Nell'ambito della Ricerca Operativa (Operations Research) uno dei principali argomenti è la cosiddetta **ottimizzazione lineare**, ossia lo studio di una classe di problemi del tipo:

$$\begin{gathered}\min / \max c^T x \\ \text{subject to} \\ Ax \lesseqgtr b\end{gathered}$$

con $x \in \mathbb{R}^n$, $c \in \mathbb{R}^n$, $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^{1 \times m}$.

In altre parole, si vogliono trovare dei valori per le componenti del vettore x tali da massimizzare (o minimizzare) il valore di una *funzione lineare* (detta **funzione obiettivo**, mentre il vettore c è chiamato **vettore dei costi**), dati una serie di vincoli espressi nel sistema di disequazioni lineari $Ax \lesseqgtr b$ (dove A è detta **matrice dei vincoli** e b **vettore dei termini noti**).

Questa classe di problemi permette di modellare un gran numero di situazioni reali in molteplici ambiti (ottimizzazione dei costi, creazione di orari, *etc.*), oltre che alcuni problemi NP-hard come il *Knapsack* o il *Vertex Cover*.

Dal punto di vista dell'algebra lineare, un problema in n varibili non è altro che uno spazio $\mathbb{R}^n$, la funzione obiettivo è una retta nello spazio, mentre i vincoli definiscono un *poliedro* nello spazio.

Utilizzando i teoremi e le tecniche dell'algebra lineare è stato possibile creare degli algoritmi per risolvere i problemi di ottimizzazione, come il **simplesso**, i quali si prestano molto bene ad essere parallelizzati (in quanto operano su matrici).

Problemi risolvibili, non risolvibili, illimitati

Dato un problema di ottimizzazione, questo può essere di tre tipi:

- avere una o più soluzioni ammissibili (*feasible*): ossia esistono uno o più vettori che moltiplicati per il vettore dei costi assegnano alla funzione obiettivo il valore massimo (o minimo possibile) e tutti i vincoli sono veri,
- non risolvibili (*infeasible*): se non esistono soluzioni
- illimitati (*unbounded*): se per una o più componenti della soluzione è possibile aumentarne (o ridurne) il valore all'infinito senza mai violare i vincoli

Forma canonica e forma standard

I problemi di massimizzazione possono essere convertiti in problemi di minimizzazione (e viceversa), così come è possibile manipolare le singole disequazioni. Queste operazioni servono a riportare i problemi in una forma che permetta di utilizzarli da parte dei solver. In particolare vengono utilizzate la forma *canonica* e la forma *standard*.

Un problema (di massimizzazione) è in forma canonica se è nella forma

$$\begin{gathered} \max c^T x \\ \text{subject to} \\ Ax \leq b \\ x \geq 0 \end{gathered}$$

Considerando che qualsiasi disequazione nella forma $\alpha x \leq y$ può essere convertita in una equazione equivalente $\alpha x + \delta = b$ definiamo la forma standard di un problema in forma canonica

$$\begin{gathered} \max d^T x \\ \text{subject to} \\ A'x = b \\ x \geq 0 \end{gathered}$$

con $A' = (A|I) \in \mathbb{R}^{m \times (n+m)}$ e $d = (c|0) \in \mathbb{R}^{n+m}$

In altre parole aggiungiamo una nuova variabile al problema (detta variabile *slack*) per ogni disequazione.

La forma standard è quella che viene utilizzata dagli algoritmi di soluzione.

Metodo del simplesso a due fasi

Il metodo del simplesso è un'algoritmo per risolvere i problemi di ottimizzazione lineari.

Si basa sul teorema secondo il quale le soluzioni (dette *soluzioni di base*) di un qualsiasi problema in forma standard sono i *vertici* del poliedro costruito sui vincoli.

Ogni vertice è individuato da una o più *soluzioni di base ammissibili*, cioè tutti i valori delle variabili in base sono maggiori o uguali a 0, mentre il valore delle altre variabili è 0. Se in base sono presenti uno o più valori uguali a 0, si dice che la base è *degenere*.

Esistono diverse implementazioni del metodo del simplesso, per questo progetto verrà implementato il cosiddetto *simplesso a due fasi con il metodo del tableau*

Cos'è il tableau

Il tableau di un simplesso è la struttura dati su cui viene eseguito l'algoritmo del simplesso.

Si tratta di una matrice composta nel seguente modo:

x_1	x_2	x_3	s_1	s_2	s_3	RHS
−2	1	-1	0	0	0	0
3	1	1	1	0	0	60
1*	-1	2	0	1	0	10
1	1	-1	0	0	1	20

La prima riga della matrice sono i coefficienti della funzione obiettivo con segno invertito, mentre il primo valore dell'ultima colonna è il valore della funzione obiettivo.

Algoritmo

1. Controllo se tutti i coefficienti del vettore dei costi sono maggiori o uguali a 0. Se sì, allora il mio problema è risolto e procedo a ricavare la soluzione, altrimenti proseguo
2. Scansiono il vettore dei costi alla ricerca della *posizione* con l'elemento minimo del vettore (C_p). Questa sarà la variabile da far entrare in base[1]
3. Costruisco il *vettore degli indicatori* dividendo il vettore dei termini noti con il vettore colonna della variabile che entra in base
4. Trovo la posizione dell'elemento minimo nel vettore degli indicatori (R_p). Questa operazione serve a individuare quale variabile deve uscire dalla base (Es. se il minimo si trova a riga 3 allora dalla base uscirà la variabile memorizzata in posizione 3 nel vettore della base)
5. Individuo il *pivot* $p = R_p \cap C_p$.
6. Siccome una variabile può essere in base solo se nel suo vettore colonna è presente un unico 1, mentre tutti gli altri componenti sono uguali a 0, procedo ad aggiornare il tableau tramite operazioni di Gauss:
   - $R_p = \frac{1}{p} R_p$
   - $R_i = R_i - \frac{C_p[i]}{p} R_p \quad \forall i = 1, \ldots, n \wedge i \neq p$

   Con $C_p[i]$ l'elemento a riga i del vettore colonna C_p.
7. Ritorno al punto 1.

[1] Si noti che è possibile unire queste prime due fasi in una sola: una volta trovato l'elemento minimo è sufficiente verificare che sia < 0, se sì allora proseguo, altrimenti termino il ciclo (se il minimo è ≥ 0, allora anche tutti gli altri elementi del vettore lo sono)

8. Una volta trovato l'ottimo ricavo la soluzione nel seguente modo:
    - Se una variabile è l'i-esima variabile di base leggo il valore di quest'ultima dall'i-esima riga del vettore dei termini noti
    - altrimenti il valore di quella variabile è pari a 0

    Inoltre leggo il valore ottimo della funzione obiettivo dal primo elemento dell'ultima colonna

Complessità Considerando che l'algoritmo non fa altro che esplorare le soluzioni di base, l'algoritmo esplora al massimo $O\binom{n+m}{m}$ (con n numero di variabili e m il numero di vincoli).

Per cui nei casi peggiori il simplesso può raggiungere complessità esponenziali. Tuttavia nel corso degli anni si è visto EMPIRICAMENTE come questa complessità venga raggiunta in pochissimi casi specifici, anzi l'algoritmo si è rivelato molto efficiente su problemi "reali", dove in media raggiunge una complessità nell'ordine di $O(mn)$, anche a confronto con altri algoritmi di soluzione (polinomiali per costruzione) come il metodo dei punti interni.

Da queste considerazioni si può concludere anche come la complessità del simplesso dipenda molto dall'istanza del problema fornito in input.

Fase 1

In questa fase viene risolto un problema ausiliario

$$\begin{gathered}\max 0x - 1y \\ \text{subject to} \\ Ax + y = b \\ x \geq 0, y \geq 0\end{gathered}$$

Allo scopo di:

- capire se il problema iniziale ha soluzione è infeasible: in questo caso il valore finale della funzione obiettivo è negativo
- capire se il problema ha una o più soluzioni degeneri: in questo caso la base ottenuta alla fine di questa fase contiene una o più variabili artificiali
- se non si presenta nessuno di questi due casi, fornire in input alla fase successiva una base ammissibile di partenza

In altre parole si tratta si inserire m nuove variabili al problema (una per vincolo). Queste sono dette *variabili artificiali*.

Ad esempio per il problema

$$\begin{gathered}\max -x_1 + x_2 \\ \text{subject to} \\ x_1 + x_2 \leq 1 \\ 3x_1 + 2x_2 \leq 6\end{gathered}$$

Il corrispondente problema ausiliario sarà

$$\max 0x_1 + 0x_2 + 0s_1 + 0s_2 - a_1 - a_2$$
$$\text{subject to}$$
$$x_1 + x_2 + s_1 + a_1 = 1$$
$$3x_1 + 2x_2 + s_2 + a_2 = 6$$

Si avrà quindi il seguente tableau

x_1	x_2	s_1	s_2	a_1	a_2	RHS
0	0	0	0	1	1	0
1	1	1	0	1	0	1
3	2	0	1	0	1	6

Con base di partenza $B = \{a_1, a_2\}$

Si noti che nonostante a_1 e a_2 siano in base, la funzione obiettivo è espressa in funzione di esse (le variabili di base devono avere come coefficiente della funzione obiettivo 0, altrimenti l'algoritmo non è corretto).

Per risolvere questo problema si sottrae alla prima riga tutte le altre (eliminazione di Gauss):

$$R_0 = R_0 - R_i \quad \forall i = 1, \ldots, m$$

x_1	x_2	s_1	s_2	a_1	a_2	RHS
-4	-3	-1	-1	0	0	-7
1	1	1	0	1	0	1
3	2	0	1	0	1	6

In questo modo la funzione obiettivo non è più espressa in termini delle variabili di base e si può proseguire a risolvere il tableau.

Alla fine di questa fase si otterrà una base ammissibile di partenza per la fase successiva (a meno di problema degenere o infeasible).

Fase 2

Terminata la prima fase si eliminano le variabili artificiali dal tableau e si sostituisce la funzione obiettivo con quella del problema originale.

Durante questa operazione può capitare lo stesso problema visto in fase 1, ossia la funzione obiettivo è espressa in funzione di qualche variabile di base. Anche in questo caso la soluzione è quella di procedere a una serie di eliminazioni di Gauss:

$$R_0 = R_0 - \frac{R_0[B[i]]}{R_i[B[i]]} R_i \quad \forall i = 1, \ldots, m$$

Con $R_0[B[i]]$ e $R_i[B[i]]$ rispettivamente il valore della prima riga e della i-esima riga nella colonna della i-esima variabile di base.

Scelte implementative e algoritmi utilizzati

Gestione della memoria

Estrazione delle colonne dalla matrice

Ricerca del pivot e test di ottimalità

Eliminazione di Gauss

Aggiornamento della tabella

Risultati sperimentali