Politecnico di Torino

Relazione Tecniche di Programmazione

Andrea Angelo Raineri - s280848

3/7/21

1 Esercizio 4

1.1 Modifiche apportate alla versione di esame

Nella versione finale del codice sono state apportate due modifiche:

- **Line 8:** Modificata la condizione all'interno del ciclo for affinché non si arrestasse prima di aver verificato l'ultima sottostringa di str1 di lunghezza pari a str2.

  `for (i = 0; i < len1 - len2; i++)`

  modificato in

  `for (i = 0; i < len1 - len2 + 1; i++)`

- **Line 18:** Modificato valore di ritorno della funzione nell'uscita anticipata in modo da restituire la posizione della sottostringa identificata e non il numero di caratteri (cnt) comuni tra le sottostringhe.

  `if (cnt > len2/2) return cnt;`

  modificato in

  `if (cnt > len2/2) return i;`

1.2 Strategia utilizzata

Definite *len1* e *len2* rispettivamente le lunghezze di str1 e str2, l'algoritmo utilizzato ricerca all'interno di ogni sottostringa di dimensione len2 contenuta in str1 il numero di caratteri combacianti con la stringa str2. Il ciclo esterno identifica ad ogni iterazione una diversa sottostringa di str1 attraverso l'indice i, il ciclo interno la confronta con la stringa str2 e salva nella variabile *cnt* il numero di caratteri rispettivamente uguali tra le stringhe. Appena viene identificata una sottostringa compatibile, ovvero in cui il numero di caratteri corrispondenti sia strettamente maggiore della metà di len2, la funzione termina anticipatamente ritornando l'indice della sottostringa di str1 valida. Se non vengono identificate stringhe valide in str1 la funzione termina con un valore di default (-1).

1.3 Strutture dati utilizzate

Quando viene chiamata la funzione vengono inizializzate 5 variabili di tipo *int*:

- **int len1 len2** contenenti le dimensioni di str1 e str2, stringhe ricevute come parametri durante la chiamata
- **int i j** utilizzati come indici per i due cicli for
- **int cnt** utilizzata per memorizzare il numero di caratteri corrispondenti durante il confronto tra stringhe

2 Esercizio 5

2.1 Modifiche apportate alla versione di esame

- **Line 11** Modificato indice posizione durante il salvataggio delle medie nel vettore avgs

  `avgs[i-j+n] += M[i][j];`

  modificato in

  `avgs[i-j+n-1] += M[i][j];`

2.2 Strategia utilizzata

L'algoritmo progettato è diviso in 3 parti principali. Vengono prima salvate all'interno del vettore avgs le somme algebriche degli elementi lungo le diverse diagonali della matrice. La matrice viene scandita in modo lineare elemento per elemento e sfruttando la proprietà degli elementi lungo le diagonali ricercate di avere differenza tra indice di riga e di colonna costante si calcola la corretta cella del vettore avgs, ognuna legata ad una delle diagonali della matrice.

Successivamente si scandisce il vettore avgs per calcolare la media aritmetica, si divide dunque ogni somma algebrica precedentemente calcolata per il numero di elementi lungo la diagonale associata ad ogni cella del vettore avgs.

Infine si scandisce nuovamente il vettore avgs contenente le medie aritmetiche lungo le diagonali alla ricerca del valore di media massimo, il cui valore viene poi restituito dalla funzione.

2.3 Strutture dati utilizzate

- **int i, j** utilizzati come indici per i cicli for
- **float maxAvg** utilizzato per memorizzare il valore di media massimo
- **float avgs[DIM]** vettore di dimensione pari a quella della matrice ricevuta in ingresso utilizzato per memorizzare i valori delle medie aritmetiche lungo le diagonali della matrice al fine di ricercarne poi il massimo

3 Esercizio 6

3.1 Modifiche apportate alla versione di esame

Non sono state apportate modifiche al codice (aggiunto main al fine di utilizzo completo)

3.2 Strategia utilizzata

La funzione legge il file di input ricevuto come parametro di chiamata fino a quando riesce a estrapolare da esso un input significativo formato da 2 coppie di interi. Dopo aver letto le coordinate di due punti l'area compresa tra essi viene salvata all'interno della matrice M, rappresentante il piano cartesiano, cambiano il valore di ogni cella dell'area da 0 a 1. Nel caso in cui un'area ricopra una cella già contenente un 1 si aggiorna un contatore per le intersezioni, il cui valore viene restituito dalla funzione alla fine della lettura del file.

(*OSS:* Le aree sono state memorizzate all'interno della matrice in modo da rispecchiare il piano cartesiano stesso)

3.3 Strutture dati utilizzate

- **int x1, x2, y1, y2** utilizzati per contenere le coordinate dei punti letti dal file di input
- **int i, j** utilizzati come indici per i cicli for
- **int area** contatore per il numero di intersezioni, inizializzato a 0
- **int M[DIM][DIM]** matrice rappresentante il piano cartesiano, inizializzata a 0