****

DIPARTIMENTO DI SCIENZE MATEMATICHE, INFORMATICHE E FISICHE

**PERIODO FRAZIONARIO**

**Implementazioni e analisi delle complessità**

Cappa Davide [151953@spes.uniud.it](mailto:151953@spes.uniud.it) 151953

Marinato Federico [157893@spes.uniud.it](mailto:157893@spes.uniud.it) 157893

Raffin Matteo [156911@spes.uniud.it](mailto:156911@spes.uniud.it) 156911

**Obiettivo**:

Implementazione e analisi della complessità del calcolo del periodo frazionario minimo tra stringhe, sviluppando il problema in due versioni algoritmiche differenti.

**Concetti teorici/legenda:**

Periodo frazionario: data una stringa s, è il più piccolo intero *p > 0* tale che ∀i = 1,…,n−p s(i) = s(i+p)

La lunghezza di una stringa s viene denotata dalla lettera n.

**Versioni algoritmiche:**

I due algoritmi implementati sono:

* *PeriodNaive*
* *PeriodSmart*

*PeriodNaive* parte da p=1, ovvero il minimo periodo ammissibile, fino ad al massimo p=n e verifica se esso è effettivamente il periodo della stringa, scandendola in modo diretto; quando ciò accade l’algoritmo termina restituendo il valore di p corrente. Per verificare la proprietà del periodo frazionario p si è andati ad utilizzare un ciclo secondario per controllare che s(i) = s(i+p) per ogni i = 1,…,n−p.

*PeriodSmart* risolve il problema sfruttando il concetto di bordo calcolato progressivamente per induzione sulle sottostringhe s[1…j] ∀ j=1,…,n per ottenere una soluzione più efficiente.

**Generazione stringhe:**

La lunghezza n è compresa in un range di valori fra 100 e 50000 con distribuzione esponenziale. Tale n dunque è definita dalla seguente funzione: n(i) = floor( A\*(B^i) ) dove i varia nell’insieme {0,…,99}, mentre A e B sono tali che n(0)=100 e n(99)=50000. I risultati ottenuti sarebbero analoghi a quelli ottenuti con n(0)=1000 e n(99)=500000, conseguenza della natura delle distribuzione dei valori.

Una stringa è generata nel seguente modo:

* caso medio: tramite l’alfabeto ternario {‘a’,’b’,’c’}. Si generano i primi q caratteri, dove q rappresenta il periodo, in modo pseudo-casuale e indipendentemente tra loro, mentre la restante parte è costruita seguendo la formula *s*(*i*)=*s(i-q)* per ogni *i*=*q*+1,…,*n*. Dunque, ad ogni lunghezza n della stringa s, q avrà un valore tra 1 ed n, ottenuto in maniera aleatoria, prima che una sequenza s venga generata.
* caso pessimo: tramite l’alfabeto binario {‘a’,’b’}, in modo che l’unico carattere diverso dagli altri sia quello in ultima posizione, per raggiungere il caso pessimo di *PeriodNaive* (*PeriodSmart* non ha una stringa per caso pessimo)*.*

**Specifiche dispositivo di calcolo:**

Il sistema di calcolo su cui si sono effettuate le misurazioni ha le seguenti specifiche: Modello: Thinkpad X260 CPU: Intel Core i7-6600U (2.50GHz) Memoria principale: 8GB

**Procedimento:**

Per l’analisi dei tempi occorre prima di tutto calcolare la risoluzione R del clock di sistema, ovvero l’intervallo minimo di tempo misurabile. Nel nostro caso la risoluzione rilevata è pari a 0.0000005 ms. Per tale valore di R il tempo minimo misurabile, ovvero il tempo di esecuzione affinché l’errore relativo generato sia minore di quello voluto, è calcolato come Tmin=R\*(1/E + 1) dove E è l’errore relativo massimo ammissibile (0.001). Per tutte le misurazioni del tempo trascorso è necessario utilizzare un clock di sistema monotono, il quale sarà richiamato due volte: rispettivamente prima e dopo l’esecuzione del calcolo del periodo frazionario con *PeriodNaive* e/o *PeriodSmart*.

Per ottenere delle misurazioni accurate nel caso medio, è stata considerata, oltre al tempo medio, anche la mediana e la deviazione standard in funzione della lunghezza della stringa n. Per ogni lunghezza n, si è andati a generare 20 sequenze diverse come campione di dati per il calcolo degli indici (ciò non si rende necessario per il caso pessimo). Per il calcolo della mediana è stato ordinato il vettore contenente i tempi delle misurazioni e si è individuato il valore mediano, mentre la deviazione standard è stata calcolata rispetto al valore medio. I tempi sono stati misurati in ms con un numero di sette cifre significative.

Per la rappresentazione grafica dei tempi di *PeriodNaive* e *PeriodSmart* sono state utilizzate due scale differenti, una lineare e una logaritmica: con la prima si vede l’andamento effettivo, mentre nella seconda, essendo le lunghezze n generate con distribuzione esponenziale, viene evidenziato meglio il comportamento asintotico.

**Caso medio:**

Dal primo grafico entrambi gli algoritmi hanno un andamento abbastanza lineare: *PeriodNaive* risulta essere leggermente più efficiente di *PeriodSmart,* pur mantenendo la stessa complessità asintotica; si potrebbe dunque ipotizzare che in generale le costanti moltiplicative del primo siano più piccole. Sono visibili in particolare dei casi in cui per certe lunghezze n delle stringhe PeriodSmart ha un tempo mediano minore di PeriodNaive, dovuto alla generazione di una particolare stringa favorevole.

Analizzando la deviazione standard non emerge una simmetria tra lo scarto positivo e negativo dal valore mediano, conferma del fatto che è stata calcolata rispetto al valore medio, indice poco robusto se presenti valori anomali.

Dal secondo grafico si evincono degli andamenti approssimativamente lineari. Infatti, non si rilevano particolari differenze tra i due*.* Rispetto al primo grafico, non si evidenziano salti significativi nei tempi mediani a causa dell’applicazione della scala logaritmica su entrambi gli assi. Ciò ha ripercussioni anche sulla deviazione standard, la quale sembra decrescere all’aumentare di n ma realmente è il contrario come si può verificare dalla scala lineare.



**Caso pessimo:**

Per i grafici a seguire, che rappresentano i casi pessimi non è stata evidenziata la deviazione standard in quanto per ogni lunghezza n la costruzione della stringa non è casuale, ma segue la descrizione data. Quindi si ha una sola stringa per ogni n e dunque una sola misurazione.

*PeriodNaive* risulta avere un andamento fortemente quadratico e questo comportamento è messo in evidenza dalla scala usata nel grafico sopra, che però va a nascondere il comportamento lineare di *PeriodSmart*.

Il grafico seguente risulta essere molto diverso rispetto a quello nel caso medio.

Per entrambi gli algoritmi si verificano andamenti approssimabili a due rette. Quest’ultime, sono di coefficiente angolare differente, in particolare su *PeriodNaive* si ha un valore circa del doppio su PeriodSmart (7.14 vs 4.79).

*PeriodSmart* è più efficiente dal punto di vista temporale, tuttavia, la sua retta ha un andamento più segmentato mentre la retta di *PeriodNaive* dopo un certo n (circa 750) ha valori che seguono più fedelmente una retta.

**Conclusione:**

Dal calcolo del periodo frazionario tra due stringhe edai grafici è emerso che la versione algoritmica *PeriodSmart* in ogni casistica ha andamento lineare; tuttavia, in esecuzione utilizza dello spazio aggiuntivo dell’ordine di n.

*PeriodNaive* èquadratico nel caso peggiore, lineare nel caso medio ma non utilizza spazio aggiuntivo, per questo motivo è tendenzialmente preferibile utilizzarlo.