UNIVERSITA’ DEGLI STUDI DI UDINE

Progetto di Algoritmi e Strutture Dati

(Prima Parte)

Autori:

Dalla Torre Josè Zito
143216
143216@spes.uniud.it

Tracanelli Alessandra
143342
143342@spes.uniud.it

Professori:

Piazza Carla
Puppis Gabriele

Indice

1 Il problema

La seguente relazione ha come obiettivo quello di implementare e analizzare i tempi medi di esecuzione degli algoritmi *PeriodNaive* e *PeriodSmart.* Entrambi hanno come scopo il calcolo del periodo frazionario minimo di una stringa s, definito come il più piccolo intero $p>0$ che soddisfa la proprietà seguente: $\forall i=1,\ldots,n-p s(i)=s(i+p)$.

Il linguaggio che abbiamo scelto e utilizzato per implementare le strutture dati e gli algoritmi, nonché lo stesso adoperato per il calcolo dei tempi di esecuzione, è C, in quanto riteniamo sia uno dei linguaggi più efficienti e veloci, avendo come contro una maggiore complessità del codice.

2 Cenni teorici

Di seguito vengono riportati i costi asintotici degli algoritmi presi in questione:

- PeriodNaive ⟶ $O(n^2)$nel caso peggiore, $\Theta(n)$ in quello medio;
- PeriodSmart ⟶ $\Theta(n)$ in ogni caso;

Facendo le opportune valutazioni, ci si aspetta che l'algoritmo PeriodSmart sia il meno efficiente tra i due dato il suo costo asintotico maggiore.

L'algoritmo PeriodNaive invece, essendo una versione ottimizzata di PeriodSmart, dovrebbe risultare più efficiente. Dal momento che nell'implementazione dell'algoritmo si è ricorsi all'utilizzo di bordo della stringa (qualunque stringa t che sia, allo stesso tempo, prefisso proprio di s e suffisso proprio di s) . Ciò ci ha permesso di ridurre il problema del calcolo del periodo frazionario minimo di s al problema del calcolo della lunghezza massima di un bordo di s.

3 Calcolo dei tempi di esecuzione

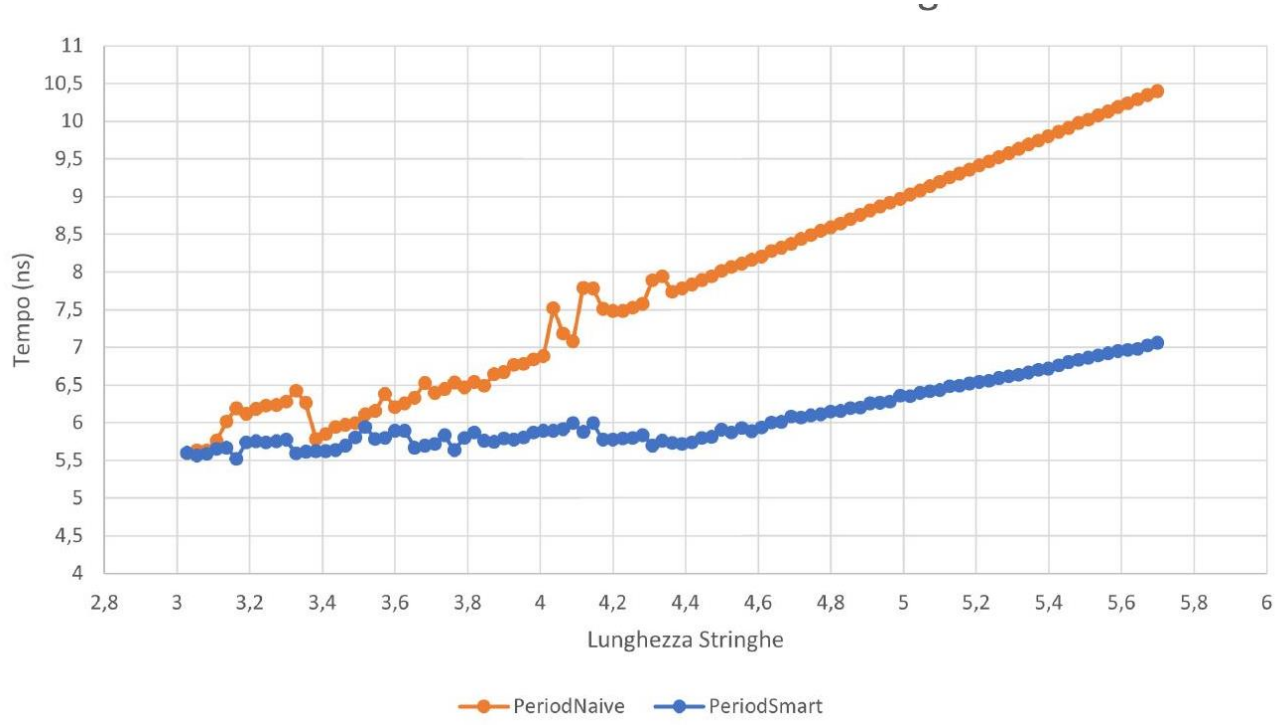
Si è scelto di utilizzare il clock offerto dalla funzione clock gettime.

Il primo parametro della funzione indica il tipo di clock, mentre il secondo parametro è un puntatore ad una struct contenente il tempo all’istante della chiamata a funzione. Il tipo di clock utilizzato (CLOCK MONOTONIC RAW)è, come suggerito dal nome stesso, di tipo monotonico; rappresenta infatti il tempo assoluto dell'orologio trascorso da un punto fisso arbitrario nel passato. Non è influenzato dalle modifiche all'orologio del sistema.

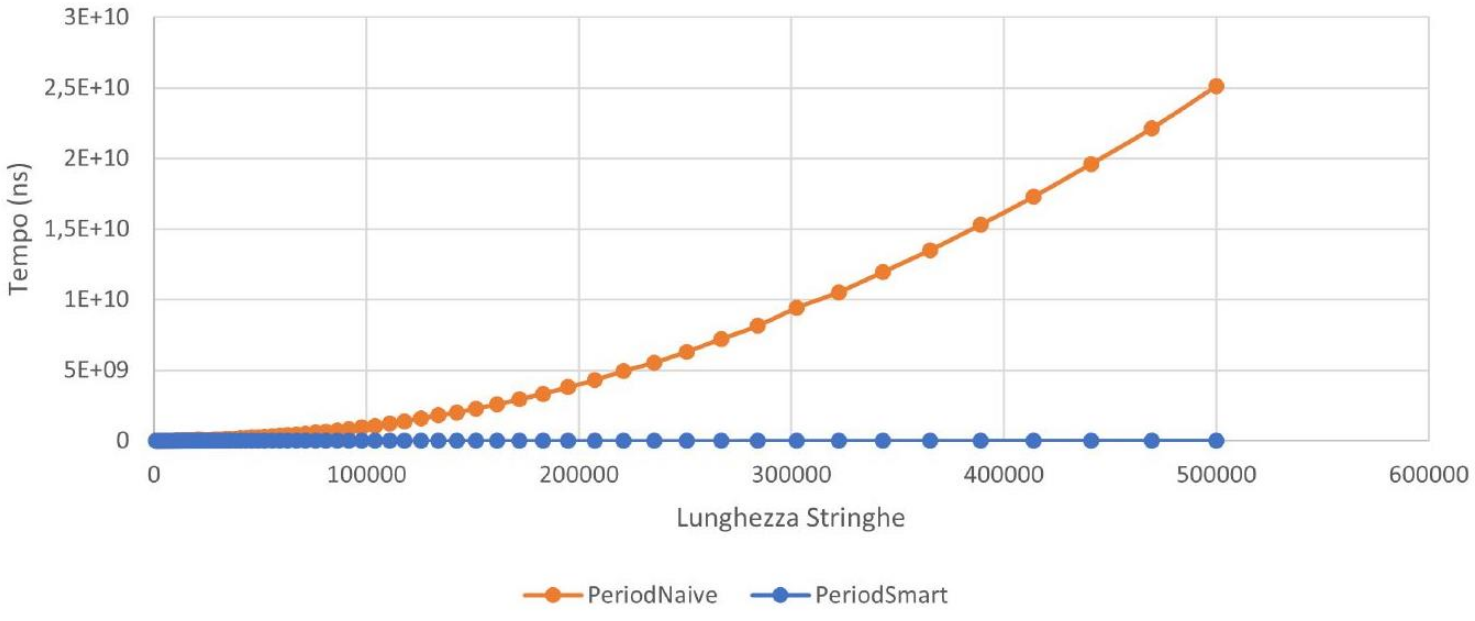
Per ottenere una misurazione della risoluzione migliore si è utilizzata una campionatura di valori che vanno da 1000 a 500000.

4 Grafici

CONFRONTO PERIODNAIVE E PERIODSMART SU GRAFICO IN SCALA LINEARE



CONFRONTO PERIODNAIVE E PERIODSMART SU GRAFICO IN SCALA LOGARITMICA



5 Considerazioni complessive

In base ai dati raccolti durante le stime dei tempi di esecuzione dei due algoritmi e analizzando i grafici ottenuti, si può notare innanzitutto come, guardando il secondo grafico realizzato su scala logaritmica, la complessità quadratica di PeriodNaive sia confermata da un andamento esponenziale della curva; per quanto riguarda PeriodSmart, dal grafico in scala lineare, si nota come all'aumentare della dimensione della stringa, l'algoritmo assuma sempre più un comportamento lineare. Questo è confermato dal fatto che il confronto che dev'essere effettuato tra le lettere di una stringa, non richiede molto tempo, a differenza di quello che accade invece per l'algoritmo Smart. In più, dalla complessità del naïve si può intendere come, sebbene sia più semplice dal punto di vista dell'implementazione, questo risulti poco efficiente, costoso ed oneroso in termini di tempo, soprattutto per stringhe di dimensioni particolarmente elevate. Da qui appunto la necessità di implementare un algoritmo smart che snellisce il problema iniziale in uno più efficiente in termini di tempo.