Analisi delle ripetizioni massimali in una stringa

Nicola De Maio, Nino Cauli

Università degli Studi di Pisa

29/04/2009

Di che cosa ci occuperemo

In questo seminario analizzeremo delle problematiche relative al numero di run in una stringa.
Per questo motivo, prima di parlare delle problmatiche principali di questa esposizione, inizieremo con lo spiegare in modo informale cosa sia una run, la sua utilità ed altri concetti utili a chiarire il background dell'argomento.

Concetto di ripetizione

Una ripetizione in una stringa w è un suo frammento in cui una sottostringa u si ripete almeno 2 volte consecutivamente:

$$w = \ldots xxx \underbrace{u \ldots u}_{\geqslant 2} xxx \ldots$$

Questa può essere intera o frazionaria a seconda che la sottostringa venga ripetuta un numero intero o meno di volte:

Ripetizione frazionaria

$w = \ldots xxxu^n vxxx \ldots$ con n intero e v prefisso di u

Concetto di run

Informalmente una run (maximal repetition) è una ripetizione non più espandibile ne' a sinistra ne' a destra.

Esempio di espandibilità

- Ripetizione espandibile a destra: $\ldots \underbrace{abcabc}_{ripetizione} a \ldots$
- Ripetizione non espandibile a destra: $\ldots \underbrace{abcabc}_{ripetizione} b \ldots$

Concetto di run 2

Esempio di run

Prendiamo la stringa

abaababaabaab

Le sue run saranno:
i prefissi "$\underbrace{abaab}\,\underbrace{abaab}\,a$"(periodo lungo 5) e "$\underbrace{aba}\,\underbrace{aba}$"(3), il suffisso "$\underbrace{aba}\,\underbrace{aba}\,ab$"(3), la sottostringa "$\underbrace{ab}\,\underbrace{ab}\,a$"(2) e le tre occorrenze di "$\underbrace{a}\,\underbrace{a}$"(1).

Utilizzo in bioinformatica

Le ripetizioni nel campo della genetica (*tandem repeats* per i biologi) sono molto importanti per determinare tratti ereditari.
Una tandem repeat di lunghezza tra 10 e 60 basi in una sequenza di DNA è detta *short tandem repeat* (STR). Analizzando la distribuzione delle STR all'interno di particolari loci è possibile ottenere un profilo genetico unico di un individuo.

Collegamento tra ripetizioni e run

Ovviamente tutte le ripetizioni presenti in una stringa saranno contenute all'interno delle sue run.
Trovare tutte le run di una stringa in tempo lineare rispetto la sua lunghezza diventa così un obbiettivo cruciale.
Come è intuibile, l'esistenza di algoritmi che riescano a trovare tutte le run di una stringa in tempo lineare rispetto alla sua lunghezza sarebbe impossibile se non fosse lineare il numero delle run stesso.

Evoluzione del limite del numero di run nella storia

Ricordo che $n = |w|$ dove w è la stringa di cui si vogliono trovare le run.

- 1981 (Crochermore): algoritmo per trovare tutte le occorrenze delle massime ripetizioni ad esponente intero in $O(n \log n)$.
- 1989 (Main): algoritmo per trovare le occorrenze più a sinistra di tutte le run in $O(n)$.
- 1997 (Iliopulos e al.): dimostrano che il numero di run nelle stringhe di Fibonacci è $O(n)$.
- 1999-2000 (Kolpakov e Kucherov): dimostrazione che in una qualsiasi stringa $|run| \leqslant cn$. Modificano l'algoritmo di Main per trovare tutte le occorrenze delle run in una stringa in $O(n)$. Congetturarono inoltre che $c = 1$
- 2006 (Rytter): Inizialmente prova che $c \leqslant 5$ utilizzando la nozione di vicinato.In seguito abbassa ancor più il bound a 3.44.

In questa dimostrazione mostreremo come questo limite è

Evoluzione del limite del numero di run nella storia

Ricordo che $n = |w|$ dove w è la stringa di cui si vogliono trovare le run.

- 1981 (Crochermore): algoritmo per trovare tutte le occorrenze delle massime ripetizioni ad esponente intero in $O(n \log n)$.
- 1989 (Main): algoritmo per trovare le occorrenze più a sinistra di tutte le run in $O(n)$.
- 1997 (Iliopulos e al.): dimostrano che il numero di run nelle stringhe di Fibonacci è $O(n)$.
- 1999-2000 (Kolpakov e Kucherov): dimostrazione che in una qualsiasi stringa $|run| \leqslant cn$. Modificano l'algoritmo di Main per trovare tutte le occorrenze delle run in una stringa in $O(n)$. Congetturarono inoltre che $c = 1$
- 2006 (Rytter): Inizialmente prova che $c \leqslant 5$ utilizzando la nozione di vicinato.In seguito abbassa ancor più il bound a 3.44.

In questa dimostrazione mostreremo come questo limite è

Evoluzione del limite del numero di run nella storia

Ricordo che $n = |w|$ dove w è la stringa di cui si vogliono trovare le run.

- 1981 (Crochermore): algoritmo per trovare tutte le occorrenze delle massime ripetizioni ad esponente intero in $O(n \log n)$.
- 1989 (Main): algoritmo per trovare le occorrenze più a sinistra di tutte le run in $O(n)$.
- 1997 (Iliopulos e al.): dimostrano che il numero di run nelle stringhe di Fibonacci è $O(n)$.
- 1999-2000 (Kolpakov e Kucherov): dimostrazione che in una qualsiasi stringa $|run| \leqslant cn$. Modificano l'algoritmo di Main per trovare tutte le occorrenze delle run in una stringa in $O(n)$. Congetturarono inoltre che $c = 1$
- 2006 (Rytter): Inizialmente prova che $c \leqslant 5$ utilizzando la nozione di vicinato.In seguito abbassa ancor più il bound a 3.44.

In questa dimostrazione mostreremo come questo limite è

Evoluzione del limite del numero di run nella storia

Ricordo che $n = |w|$ dove w è la stringa di cui si vogliono trovare le run.

- 1981 (Crochermore): algoritmo per trovare tutte le occorrenze delle massime ripetizioni ad esponente intero in $O(n \log n)$.
- 1989 (Main): algoritmo per trovare le occorrenze più a sinistra di tutte le run in $O(n)$.
- 1997 (Iliopulos e al.): dimostrano che il numero di run nelle stringhe di Fibonacci è $O(n)$.
- 1999-2000 (Kolpakov e Kucherov): dimostrazione che in una qualsiasi stringa $|run| \leqslant cn$. Modificano l'algoritmo di Main per trovare tutte le occorrenze delle run in una stringa in $O(n)$. Congetturarono inoltre che $c = 1$
- 2006 (Rytter): Inizialmente prova che $c \leqslant 5$ utilizzando la nozione di vicinato.In seguito abbassa ancor più il bound a 3.44.

In questa dimostrazione mostreremo come questo limite è

Evoluzione del limite del numero di run nella storia

Ricordo che $n = |w|$ dove w è la stringa di cui si vogliono trovare le run.

- 1981 (Crochermore): algoritmo per trovare tutte le occorrenze delle massime ripetizioni ad esponente intero in $O(n \log n)$.
- 1989 (Main): algoritmo per trovare le occorrenze più a sinistra di tutte le run in $O(n)$.
- 1997 (Iliopulos e al.): dimostrano che il numero di run nelle stringhe di Fibonacci è $O(n)$.
- 1999-2000 (Kolpakov e Kucherov): dimostrazione che in una qualsiasi stringa $|run| \leqslant cn$. Modificano l'algoritmo di Main per trovare tutte le occorrenze delle run in una stringa in $O(n)$. Congetturarono inoltre che $c = 1$
- 2006 (Rytter): Inizialmente prova che $c \leqslant 5$ utilizzando la nozione di vicinato.In seguito abbassa ancor più il bound a 3.44.

In questa dimostrazione mostreremo come questo limite è

Evoluzione del limite del numero di run nella storia

Ricordo che $n = |w|$ dove w è la stringa di cui si vogliono trovare le run.

- 1981 (Crochermore): algoritmo per trovare tutte le occorrenze delle massime ripetizioni ad esponente intero in $O(n \log n)$.
- 1989 (Main): algoritmo per trovare le occorrenze più a sinistra di tutte le run in $O(n)$.
- 1997 (Iliopulos e al.): dimostrano che il numero di run nelle stringhe di Fibonacci è $O(n)$.
- 1999-2000 (Kolpakov e Kucherov): dimostrazione che in una qualsiasi stringa $|run| \leqslant cn$. Modificano l'algoritmo di Main per trovare tutte le occorrenze delle run in una stringa in $O(n)$. Congetturarono inoltre che $c = 1$
- 2006 (Rytter): Inizialmente prova che $c \leqslant 5$ utilizzando la nozione di vicinato.In seguito abbassa ancor più il bound a 3.44.

In questa dimostrazione mostreremo come questo limite è

Definizioni di base

Ecco alcune definizioni che ci saranno utili nel seguito della presentazione:

- Una stringa w ha periodo p sse $w[i] = w[i+p]$, per $1 \leqslant i \leqslant |w| - p$.
- Il più piccolo periodo di w è chiamato *il periodo* di w.
- Il rapporto tra la lunghezza di w ed il suo periodo è chiamato esponente di w (se w ha periodo p allora $w \equiv x^{\frac{|w|}{p}}$ per qualche x prefisso di w).
- Una stringa è *primitiva* se non può essere scritta come potenza intera (2 o più) di un'altra stringa.

Definizioni di base

Ecco alcune definizioni che ci saranno utili nel seguito della presentazione:

- Una stringa w ha periodo p sse $w[i] = w[i+p]$, per $1 \leqslant i \leqslant |w| - p$.
- Il più piccolo periodo di w è chiamato *il periodo* di w.
- Il rapporto tra la lunghezza di w ed il suo periodo è chiamato esponente di w (se w ha periodo p allora $w \equiv x^{\frac{|w|}{p}}$ per qualche x prefisso di w).
- Una stringa è *primitiva* se non può essere scritta come potenza intera (2 o più) di un'altra stringa.

Definizioni di base

Ecco alcune definizioni che ci saranno utili nel seguito della presentazione:

- Una stringa w ha periodo p sse $w[i] = w[i+p]$, per $1 \leqslant i \leqslant |w| - p$.
- Il più piccolo periodo di w è chiamato *il periodo* di w.
- Il rapporto tra la lunghezza di w ed il suo periodo è chiamato esponente di w (se w ha periodo p allora $w \equiv x^{\frac{|w|}{p}}$ per qualche x prefisso di w).
- Una stringa è *primitiva* se non può essere scritta come potenza intera (2 o più) di un'altra stringa.

Definizioni di base

Ecco alcune definizioni che ci saranno utili nel seguito della presentazione:

- Una stringa w ha periodo p sse $w[i] = w[i+p]$, per $1 \leqslant i \leqslant |w| - p$.
- Il più piccolo periodo di w è chiamato *il periodo* di w.
- Il rapporto tra la lunghezza di w ed il suo periodo è chiamato esponente di w (se w ha periodo p allora $w \equiv x^{\frac{|w|}{p}}$ per qualche x prefisso di w).
- Una stringa è *primitiva* se non può essere scritta come potenza intera (2 o più) di un'altra stringa.

Concetto di run

Definizione di run e di altri concetti collegati:

- È detto run l'intervallo $[i \ldots j]$ di una stringa w tale che:
  1. la sottostringa $w[i \ldots j]$ abbia esponente del suo periodo minimo $\geqslant 2$.
  2. le sottostringe $w[i-1 \ldots j]$ e $w[i \ldots j+1]$ abbiano il periodo minimo di lunghezza > rispetto a quello di $w[i \ldots j]$.
- La generatrice di una run di periodo p è il suo prefisso lungo p
- Le *square* sono stringhe con esponente = 2.
- Definiamo *la square* di una run di periodo p il suo prefisso lungo $2p$.
- Presa una run $w[i \ldots j]$ di periodo p, chiamiamo *l'inizio* della run i e il suo *centro* $c = i + p$.

Concetto di run

Definizione di run e di altri concetti collegati:

- È detto run l'intervallo $[i \ldots j]$ di una stringa w tale che:
  1. la sottostringa $w[i \ldots j]$ abbia esponente del suo periodo minimo $\geqslant 2$.
  2. le sottostringe $w[i-1 \ldots j]$ e $w[i \ldots j+1]$ abbiano il periodo minimo di lunghezza $>$ rispetto a quello di $w[i \ldots j]$.
- La generatrice di una run di periodo p è il suo prefisso lungo p
- Le *square* sono stringhe con esponente = 2.
- Definiamo *la square* di una run di periodo p il suo prefisso lungo $2p$.
- Presa una run $w[i \ldots j]$ di periodo p, chiamiamo *l'inizio* della run i e il suo *centro* $c = i + p$.

Concetto di run

Definizione di run e di altri concetti collegati:

- È detto run l'intervallo $[i \ldots j]$ di una stringa w tale che:
  1. la sottostringa $w[i \ldots j]$ abbia esponente del suo periodo minimo $\geqslant 2$.
  2. le sottostringe $w[i-1 \ldots j]$ e $w[i \ldots j+1]$ abbiano il periodo minimo di lunghezza $>$ rispetto a quello di $w[i \ldots j]$.
- La generatrice di una run di periodo p è il suo prefisso lungo p
- Le *square* sono stringhe con esponente = 2.
- Definiamo *la square* di una run di periodo p il suo prefisso lungo $2p$.
- Presa una run $w[i \ldots j]$ di periodo p, chiamiamo *l'inizio* della run i e il suo *centro* $c = i + p$.

Concetto di run

Definizione di run e di altri concetti collegati:

- È detto run l'intervallo $[i \ldots j]$ di una stringa w tale che:
  1. la sottostringa $w[i \ldots j]$ abbia esponente del suo periodo minimo $\geqslant 2$.
  2. le sottostringe $w[i-1 \ldots j]$ e $w[i \ldots j+1]$ abbiano il periodo minimo di lunghezza $>$ rispetto a quello di $w[i \ldots j]$.
- La generatrice di una run di periodo p è il suo prefisso lungo p
- Le *square* sono stringhe con esponente = 2.
- Definiamo *la square* di una run di periodo p il suo prefisso lungo $2p$.
- Presa una run $w[i \ldots j]$ di periodo p, chiamiamo *l'inizio* della run i e il suo *centro* $c = i + p$.

Lemma di Fine e Wilf

Enunciamo ora un lemma ed una proprietà che ci saranno utili nelle dimostrazioni future:

Lemma della periodicità di Fine e Wilf

Se una stringa w ha come periodi p e q, e $|w| \geqslant p + q - \gcd(p, q)$ allora w ha anche periodo $\gcd(p, q)$.

Proprietà di sincronizzazione

Se w è primitiva allora apparirà come fattore di ww solo come prefisso o suffisso.

Ricerca del limite

Vediamo ora come sia possibile dimostrare il

Teorema sul numero di run

In una stringa w di lunghezza n il numero di run è al più $1.6n$.

Concetto generale

1. Distribuiremo tutte le run in gruppi a seconda della lunghezza del loro periodo.
2. Troveremo il numero massimo di run appartenenti ad ogni gruppo e li sommeremo per ottenere il limite.

δ-vicine e δ-run

Concetto importante per definire il limite di run è quello di gruppi δ-vicine:

Definizione di δ-run

Se per una run di periodo p vale $2\delta \leqslant p \leqslant 3\delta$, allora può essere definita δ-run.

Definizione della relazione δ-vicine

Per ogni $\delta > 0$, due run con square x^2 e y^2 sono δ-vicine se:

1. Sono delle δ-run.
2. $|c_x - c_y| \leqslant \delta$.

Scelta dei δ

Per ottenere una sequenza $\delta_1, \delta_2, \ldots$ valori di δ che coprano tutti i possibili periodi bisogna sceglierli in modo tale che i corrispondenti intervalli $[2\delta_i \ldots 3\delta_i]$ coprano colletivamente i reali $\geqslant 1$.
Una buona scelta per i δ_i può essere la seguente:

$$\delta_i = \frac{1}{2}(\frac{3}{2})^i,\ i \geqslant 0$$

In questo modo l'insieme di intervalli ottenuto sarà infatti:

$$[1 \ldots \tfrac{3}{2}], [\tfrac{3}{2} \ldots (\tfrac{3}{2})^2], [(\tfrac{3}{2})^2 \ldots (\tfrac{3}{2})^3] \ldots$$

Numero di δ-run per stringa

Il raggiungimento del limite di 1.6 si basa fortemente sul seguente teorema che più avanti dimostreremo:

Teorema sul numero di δ-run

Per ogni δ, divisa la stringa in intervalli lunghi δ, sarà contenuto al più mediamente 1 centro di una δ-run in ogni intervallo.

Da questo possiamo derivare il corollario:

Corollario sul numero di δ-run

Il numero di δ-run in una stringa lunga n è al più $\frac{n}{\delta}$.

Microrun e macrorun

Per riuscire ad ottenere il limite sopracitato è necessario suddividere l'insieme delle run in 2 gruppi:

1. Tutte le run con lunghezza del periodo $|x| \leqslant 9$ (microrun).
2. Tutte le run con lunghezza del periodo $|x| > 9$

Questa suddivisione è stata fatta dal momento che riusciremo a dimostrare più avanti che il numero di microrun all'interno di una stringa è al più n.

Limite

A questo punto possiamo affermare che le run all'interno di una stringa lunga n sono al massimo:

$$n + |run_{\geqslant 10}|$$

Possiamo scegliere i δ_i in modo da coprire con gli intervalli $[2\delta_i \dots 3\delta_i]$ tutti i reali $\geqslant 10$ esattamente come fatto in precedenza:

$$\delta_i = \frac{10}{2}(\frac{3}{2})^i,\ i \geqslant 0$$

In questo modo tenendo conto del Teorema sul numero di δ-run:

Limite

$$\underbrace{n}_{|run_{<10}|} + \underbrace{\sum_{i=0}^{\infty} \frac{n}{\delta_i}}_{|run_{\geqslant 10}|} = n + (\frac{2}{10}\sum_{i=0}^{\infty}(\frac{2}{3})^i)n = 1.6n$$

Dimostrazioni

Perchè il limite sia valido ci restano da dimostrare 2 assunzioni:

assunzioni da dimostrare

1. Per ogni δ il numero delle δ-run in una stringa lunga n è al più $\frac{n}{\delta}$.
2. Il numero delle run con periodo di lunghezza < 10 in una stringa lunga n sono al più n.

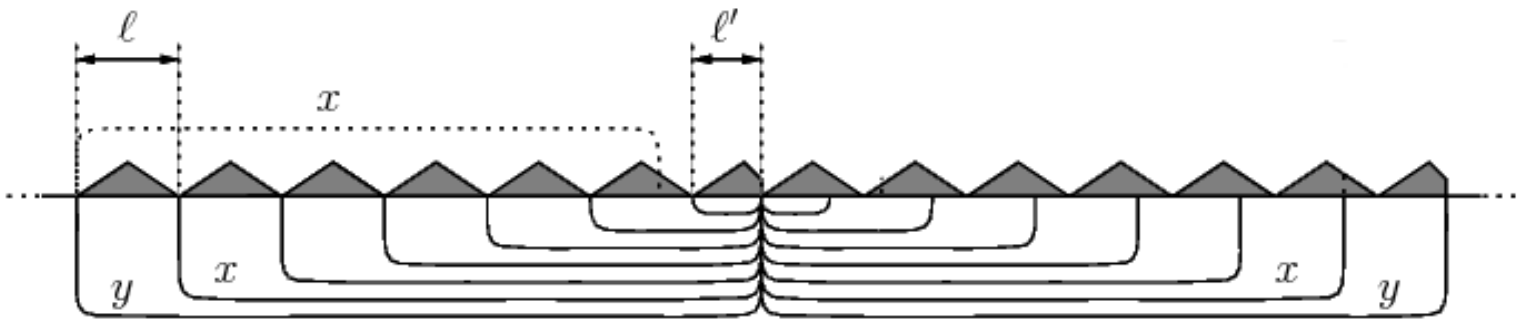
Dimostrazione macrorun

Assumiamo di avere tre run con squares x^2, y^2, z^2 e con periodi lunghi rispettivamente $|x| = p \leqslant |y| = q \leqslant |z| = r$.
Chiamiamo inoltre δ-intervalli l'insieme di intervalli lunghi δ in cui suddividiamo la srtinga.
Per dimostrare la prima assunzione (teorema sul numero di δ-run) dovremo trovare un numero di δ_i-intervalli in cui non cade nessun centro di δ_i-run tale che riesca a coprire il numero di δ-run troppo vicine tra loro

Casi che analizzeremo

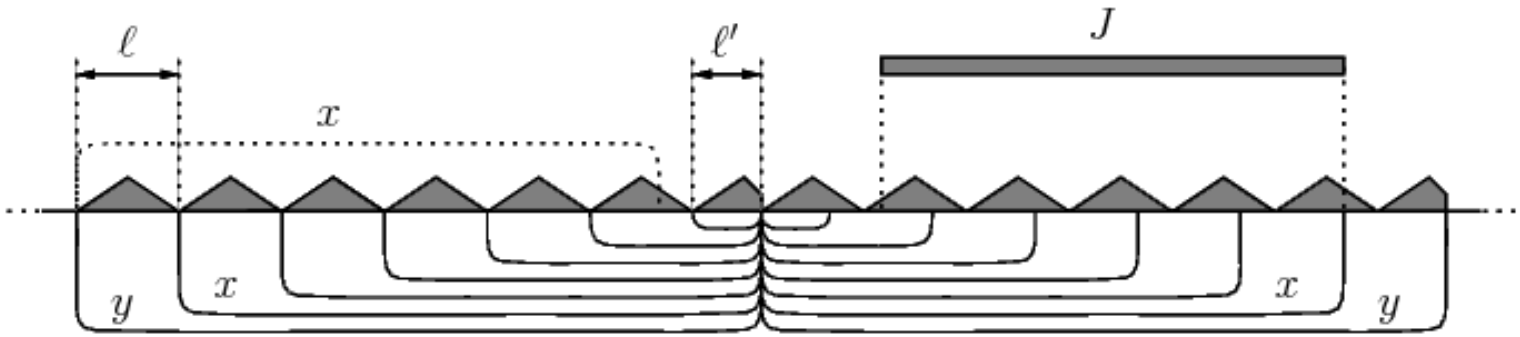
1. Caso x^2 e y^2 δ-vicine concentriche.
2. Caso generico x^2 e y^2 δ-vicine.

Dimostrazione run concentrici



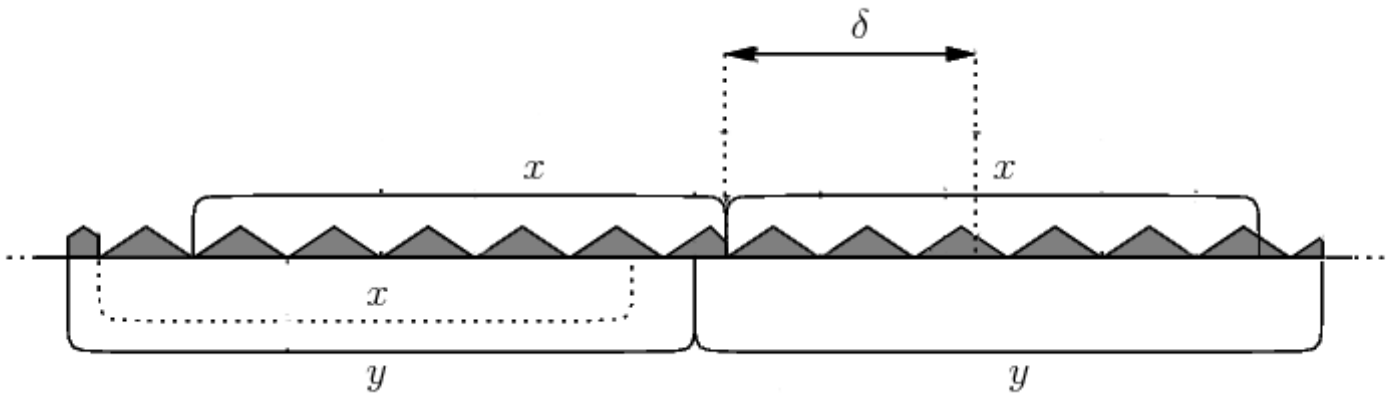
- Presi x^2 e y^2 concentrici, esiste un piccolo segmento lungo $l = q - p$ che è periodo sia di x che di y.
- Sia x che y finiscono con un segmento l' prefisso di l.
- Con questa configurazione esistono altre $h = \lceil \frac{q}{l} \rceil$ run concentriche che prenderemo in considerazione.
- Nel caso $h < 7$ non posso esistere 3 di queste run δ-vicine tra di loro, questo caso lo analizzeremo nella prossima dimostrazione.
- Qui prenderemo in considerazione solo $h \geqslant 6$.

Dimostrazione run concentrici



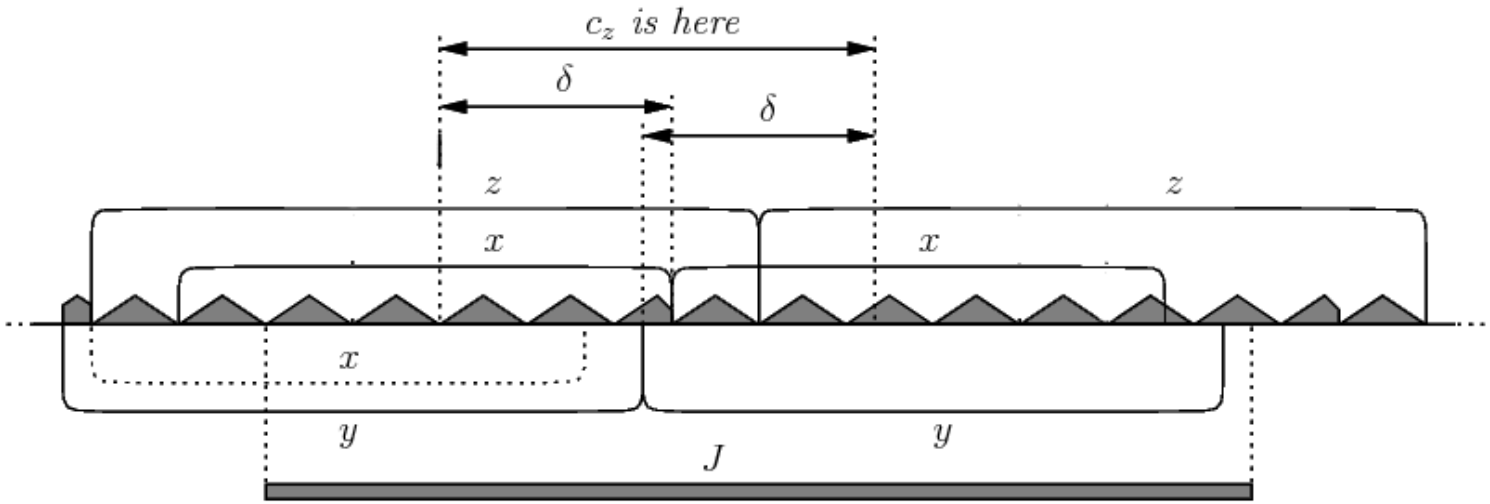
- Essendo il minimo numero di δ che coprono le h run almeno 3, allora dobbiamo trovare $h-3$ δ_i-intervalli vuoti (senza centri di δ_i-run) che coprano le run in eccesso.
- Prendiamo δ_{i_0} tale che $\frac{l}{2} \leqslant \delta_{i_0} \leqslant \frac{3}{4} l$.
- Prendendo un intervallo J come in figura si può dimostrare che al suo interno non cade nessun centro di δ_{i_0}-run.
- Essendo $|J| \geqslant (h-2)\delta_{i_0}$ per costruzione, esso coprirà almeno $(h-3)$ δ_{i_0}-intervalli C.V.D..

Dimostrazione altri casi



- Analiziamo il caso $i_y < i_x < c_y \leqslant c_x < e_x < e_y$ come esempio generale essendo molto simile a tutti gli altri.
- Anche in questo caso come in precedenza esiste un piccolo segmento lungo $l = q - p$ che è periodo sia di x che di un lungo suffisso di y.
- Se non esiste un'altra run δ-vicina ad x^2 e y^2 allora si può dimostrare che il δ-intervallo successivo alle 2 run δ-vicine è vuoto, quindi il teorema sul numero di δ-run continua a valere.

Dimostrazione altri casi



- Nel caso esista una terza square z^2 δ-vicina a x^2 e y^2 allora questa può essere solamente quella rappresentata in figura con inizio in $i_x - l$.
- Se prendiamo J come in figura lungo almeno 4δ, qusto coprirà almeno 3 δ-intervalli.

Dimostrazione altri casi

- Si può dimostrare che non esistono altre run con $c \neq c_x$ δ-vicine alle 3 trovate in precedenza, quindi in questo caso vengono coperte le 3 square δ-vicine.
- Può esistere per $c = c_x$, ma si può utilizzare il ragionamento fatto in precedenza per gestirla essendoci 3 run concentriche δ-vicine.

microrun

Per i Run di periodo più piccolo viene effettuata un'analisi a parte.
consideriamo i Run con periodo di lunghezza fino a 9.
per ogni posizione i dentro la stringa w consideriamo i Run con centro in i e ammortizziamo il loro numero contando i Run con centro in $[i-j,\ldots,i]$ dove j sarà al più 4.
ciò che si dimostra è che in media abbiamo al più un centro di microrun per posizione.
Diremo che c'è un microrun di periodo p e centro in i dicendo che abbiamo p in i. indicheremo l'insieme dei periodi dei run di centro i con $C(i) = \{p|p \text{ è in } i\}$.

microrun

Per i Run di periodo più piccolo viene effettuata un'analisi a parte.
consideriamo i Run con periodo di lunghezza fino a 9.
per ogni posizione i dentro la stringa w consideriamo i Run con centro in i e ammortizziamo il loro numero contando i Run con centro in $[i-j,\ldots,i]$ dove j sarà al più 4.
ciò che si dimostra è che in media abbiamo al più un centro di microrun per posizione.
Diremo che c'è un microrun di periodo p e centro in i dicendo che abbiamo p in i. indicheremo l'insieme dei periodi dei run di centro i con $C(i) = \{p|p \text{ è in } i\}$.

Lemma di inclusione

Il seguente lemma è lo strumento con il quale si effettua l'analisi:

Lemma di inclusione dei microrun

Data una stringa w , due interi $p > l > 0$, e definito $h = \lceil \frac{p}{l} \rceil$,allora

1. Se w ha (ripetizioni di) periodi p e $p - l$ rispettivamente in j e i con $|i - j| \leqslant l$, allora w ha anche i periodi $p - kl$ con $2 \leqslant k \leqslant h - 1$ in i.
2. Se w ha (run di periodi) p e $p - l$ rispettivamente in j e i con $|i - j| \leqslant l$, allora w ha anche $p - kl$ con $2 \leqslant k \leqslant h - 3$ in i.

Un altro risultato (questa volta banale) che serve nell'analisi è la seguente:

Nota

se w ha p in i allora w non ha p in $[i - p, \ldots, i + p]$

Lemma di inclusione

Il seguente lemma è lo strumento con il quale si effettua l'analisi:

Lemma di inclusione dei microrun

Data una stringa w , due interi $p > l > 0$, e definito $h = \lceil \frac{p}{l} \rceil$,allora

1. Se w ha (ripetizioni di) periodi p e $p - l$ rispettivamente in j e i con $|i - j| \leqslant l$, allora w ha anche i periodi $p - kl$ con $2 \leqslant k \leqslant h - 1$ in i.
2. Se w ha (run di periodi) p e $p - l$ rispettivamente in j e i con $|i - j| \leqslant l$, allora w ha anche $p - kl$ con $2 \leqslant k \leqslant h - 3$ in i.

Un altro risultato (questa volta banale) che serve nell'analisi è la seguente:

Nota

se w ha p in i allora w non ha p in $[i - p, \ldots, i + p]$

applicazioni lemma di inclusione

questi due risultati vengono usati in 2 modi: data una posizione i della stringa w riduciamo lo spazio delle possibili configurazioni di $C(i)$.
Se per esempio $1, 2, 3 \notin C(i)$ ma $5 \in C(i)$, il lemma di inclusione ci dice che $C(i) = \{5\}$.
a priori le possibili configurazioni di $C(i)$ sarebbero 2^9. in questo modo vengono ridotte a 24.
Per ognuna delle configurazioni rimaste di $C(i)$ viene poi svolta un'analisi dettagliata sul contorno della posizione i per trovarne un $[i - j, \ldots, i]$ con in media un run per posizione.
Ad esempio se $C(i) = \{1, 3\}$ allora $w[i - 4, \ldots, i + 2] = \bar{a}abaaaba$ e $C(i - 1)$ puó avere periodi dal 5 in su.
nel caso abbia 5 allora $w[i - 7, \ldots, i + 3] = \bar{b}aababaabab$ e allora $C(i - 1) = \{5\}$ e $C(i - 2) = \emptyset$.

applicazioni lemma di inclusione

questi due risultati vengono usati in 2 modi: data una posizione i della stringa w riduciamo lo spazio delle possibili configurazioni di $C(i)$.
Se per esempio $1, 2, 3 \notin C(i)$ ma $5 \in C(i)$, il lemma di inclusione ci dice che $C(i) = \{5\}$.
a priori le possibili configurazioni di $C(i)$ sarebbero 2^9. in questo modo vengono ridotte a 24.
Per ognuna delle configurazioni rimaste di $C(i)$ viene poi svolta un'analisi dettagliata sul contorno della posizione i per trovarne un $[i - j, \ldots, i]$ con in media un run per posizione.
Ad esempio se $C(i) = \{1, 3\}$ allora $w[i - 4, \ldots, i + 2] = \bar{a}abaaaba$ e $C(i - 1)$ puó avere periodi dal 5 in su.
nel caso abbia 5 allora $w[i - 7, \ldots, i + 3] = \bar{b}aababaabab$ e allora $C(i - 1) = \{5\}$ e $C(i - 2) = \emptyset$.

Lemma sul numero dei microrun

Lavorando così per tutte le possibili configurazioni di $C(i)$ otteniamo il preannunciato

Lemma sul numero dei microrun

in una stringa w di lunghezza n il numero di run con periodo al piú 9 è maggiorato da n.

Questo conclude la dimostrazione del Teorema sul numero di run.
Questo modo di procedere puó essere automatizzato e utilizzato per i microrun di periodo piú grande di 9.
Una strada per la soluzione della congettura potrebbe essere l'estensione del Lemma di inclusione ai run di qualsiasi periodo, oppure ad un rafforzamento in termini di costante del Lemma sul numero di microrun.

Lemma sul numero dei microrun

Lavorando così per tutte le possibili configurazioni di $C(i)$ otteniamo il preannunciato

Lemma sul numero dei microrun

in una stringa w di lunghezza n il numero di run con periodo al piú 9 è maggiorato da n.

Questo conclude la dimostrazione del Teorema sul numero di run.
Questo modo di procedere puó essere automatizzato e utilizzato per i microrun di periodo piú grande di 9.
Una strada per la soluzione della congettura potrebbe essere l'estensione del Lemma di inclusione ai run di qualsiasi periodo, oppure ad un rafforzamento in termini di costante del Lemma sul numero di microrun.

limite dal basso

Abbiamo visto che il numero di run in una stringa di lunghezza n non arriva ad $1.6n$, e vedremo che questo bound può essere migliorato senza troppi sforzi
Ma fin dove si può arrivare? i test al computer suggeriscono meno di n, e anche se n è attualmente solo una congettura, in pochi sono convinti che esso sia il meglio ottenibile.
Ora mostriamo che non si può scendere sotto il valore

$$\alpha = \frac{3}{1+\sqrt{5}} \approx 0.927.$$

limite dal basso

Definiamo $\rho(w)$ il numero di run nella stringa w.
chiamiamo $\rho(n)$ il massimo valore di $\rho(w)$ al variare di w tra le stringhe di lunghezza n.
Abbiamo finora dimostrato che $\rho(n) \leq 1.6n$.

Non si sa se la successione $\frac{\rho(n)}{n}$ ammetta limite, sappiamo anzi che essa non è crescente.

Però $\frac{\rho(n)}{n}$ è limitata dall'alto da 1.6 e banalmente si vede che è limitata dal basso da 0.5. Ciò che mostriamo ora è che $\limsup \frac{\rho(n)}{n} \geq \alpha$, ma il ragionamento che faremo si può approfondire per arrivare a dire che $\liminf \frac{\rho(n)}{n} \geq \alpha$.

limite dal basso

Definiamo $\rho(w)$ il numero di run nella stringa w.
chiamiamo $\rho(n)$ il massimo valore di $\rho(w)$ al variare di w tra le stringhe di lunghezza n.
Abbiamo finora dimostrato che $\rho(n) \leq 1.6n$.

Non si sa se la successione $\frac{\rho(n)}{n}$ ammetta limite, sappiamo anzi che essa non è crescente.

Però $\frac{\rho(n)}{n}$ è limitata dall'alto da 1.6 e banalmente si vede che è limitata dal basso da 0.5. Ciò che mostriamo ora è che $\limsup \frac{\rho(n)}{n} \geq \alpha$, ma il ragionamento che faremo si può approfondire per arrivare a dire che $\liminf \frac{\rho(n)}{n} \geq \alpha$.

La successione di stringhe

Costruiamo una successione di stringhe binarie x_i tali che $\lim \frac{\rho(x_i)}{|x_i|} = \alpha$. $x_1 = 1$ e ogni stringa x_i viene ottenuta dalla precedente x_{i-1} sostituendo ad ogni lettera una stringa, esattamente

- 0 è sostituito da 010010.
- 1 è sostituito da 101101.

come si vede ognuna di queste stringhe ci fornisce 2 run.

La successione di stringhe

Ora però queste sottostringhe vengono allacciate l'una all'altra in maniera non banale, ossia:

- 010010,010010 diventa 010010,10010 ,si perde una cifra nel mezzo e si guadagnano 2 run. 101101,101101 viene trattata in modo complementare.
- 010010,101101 diventa 01001,01101 , si cancellano le 2 cifre nel mezzo e si guadagna 1 run. 101101,010010 viene trattata allo stesso modo.

le stime sulla sottosuccessione

Oltre a farci guadagnare dei run, questo modo di procedere conserva tutti i run che avevamo nella stringa precedente.
Si dimostra che $|x_{i+1}| = 4|x_i| + |x_{i-1}| + 2$.
Da questo possiamo calcolare $|x_i|$.
Abbiamo già detto che i run della stringa precedente si conservano, inoltre se ne aggiungono 3 per ogni cifra della stringa precedente, ottenendo $\rho(x_{i+1}) = \rho(x_i) + 3|x_i| - 1$.
A partire da questi dati incrementali si vede che esiste il limite $\lim \frac{\rho(x_i)}{|x_i|} = \alpha$.

le stime sulla sottosuccessione

Oltre a farci guadagnare dei run, questo modo di procedere conserva tutti i run che avevamo nella stringa precedente.
Si dimostra che $|x_{i+1}| = 4|x_i| + |x_{i-1}| + 2$.
Da questo possiamo calcolare $|x_i|$.
Abbiamo già detto che i run della stringa precedente si conservano, inoltre se ne aggiungono 3 per ogni cifra della stringa precedente, ottenendo $\rho(x_{i+1}) = \rho(x_i) + 3|x_i| - 1$.
A partire da questi dati incrementali si vede che esiste il limite $\lim \frac{\rho(x_i)}{|x_i|} = \alpha$.

Esponenti dei microrun

Dimostriamo ora il seguente

Teorema della somma degli esponenti

La somma degli esponenti dei run in una stringa w di lunghezza n è limitato da $5.6n$.

Come abbiamo giá fatto in precedenza, per dimostrare la stima voluta spezziamo i calcoli in due: da una parte consideriamo i microrun, con il seguente

Lemma sugli esponenti dei microrun

La somma degli esponenti dei run di periodo al piú 4 in una stringa w di lunghezza n è limitato da $2n$.

Esponenti dei microrun

Dimostriamo ora il seguente

Teorema della somma degli esponenti

La somma degli esponenti dei run in una stringa w di lunghezza n è limitato da $5.6n$.

Come abbiamo giá fatto in precedenza, per dimostrare la stima voluta spezziamo i calcoli in due: da una parte consideriamo i microrun, con il seguente

Lemma sugli esponenti dei microrun

La somma degli esponenti dei run di periodo al piú 4 in una stringa w di lunghezza n è limitato da $2n$.

Periodi maggiori

Per i run di periodo maggiore utilizziamo invece il Lemma di Fine e Wilf. infatti da esso deduciamo che due δ-run non possono sovrapporsi per piú di 2.5 volte il periodo di uno dei due.
Da Ció due qualsiasi δ-run x^{α} e y^{β} di generatrici x e y non possono sovrapporsi in nessuna parte dei loro suffissi $x^{\alpha-2.5}$ e $y^{\beta-2.5}$,se questi sono definiti.
Ossia, per un fissato δ, la somma delle lunghezze dei suffissi dei δ-run è al piú n.

somma degli esponenti 2

Per ogni singolo δ-run abbiamo

$$\alpha - 2.5 = \frac{|x^{\alpha-2.5}|}{|x|} \leq \frac{|x^{\alpha-2.5}|}{2\delta}. \text{ e quindi } \alpha \leqslant 2.5 + \frac{|x^{\alpha-2.5}|}{2\delta}$$

quindi sommando per tutti i δ-run, che ricordiamo essere meno di $\frac{n}{\delta}$, abbiamo che la somma degli esponenti dei δ-run è maggiorata da

$$\sum \alpha_j \leqslant 2.5\frac{n}{\delta} + \frac{n}{2\delta}$$

somma degli esponenti 2

Per ogni singolo δ-run abbiamo

$$\alpha - 2.5 = \frac{|x^{\alpha-2.5}|}{|x|} \leq \frac{|x^{\alpha-2.5}|}{2\delta}. \text{ e quindi } \alpha \leqslant 2.5 + \frac{|x^{\alpha-2.5}|}{2\delta}$$

quindi sommando per tutti i δ-run, che ricordiamo essere meno di $\frac{n}{\delta}$, abbiamo che la somma degli esponenti dei δ-run è maggiorata da

$$\sum \alpha_j \leqslant 2.5\frac{n}{\delta} + \frac{n}{2\delta}$$

somma degli esponenti 3

Ora dobbiamo sommare per tutti i δ associati ai periodi almeno 5, e aggiungere $2n$ relativi ai periodi dal 4 in giú. Otteniamo $\delta_i = \frac{5}{2}(\frac{3}{2})^i, i \geq 0$ e la stima risulta

$$\underbrace{2n}_{\textit{microrun}} + \overbrace{\sum_{i=0}^{\infty}(2.5\frac{n}{\delta_i} + \frac{n}{2\delta_i})}^{\textit{macrorun}} = 2n + (3\tfrac{2}{5}\sum_{i=0}^{\infty}(\tfrac{2}{3})^i)n = 5.6n.$$

Miglioramento

Le due stime mostrate, sul numero di run e sulla somma degli esponenti, possono essere lievemente migliorate:
abbiamo fatto variare i δ in modo tale che $\bigcup_{i \in \mathbb{N}} [2\delta, 3\delta]$ fosse una semiretta di $\mathbb{R}$.
Possiamo scegliere i δ in modo tale che ricoprano meno, ma contengano comunque tutti i valori interi maggiori di quello fissato.
Ad esempio nel Teorema sul numero di δ-run dobbiamo ricoprire tutti i periodi dal 10 in su. possiamo allora scegliere:

- $\delta_0 = \frac{10}{2}$ che ricopre i periodi da 10 a 15.
- $\delta_1 = \frac{16}{2}$ che ricopre da 16 a 24.

E cosi' via, ma il risparmio che si ottiene non raggiunge gli $0.05n$.

Miglioramento

Le due stime mostrate, sul numero di run e sulla somma degli esponenti, possono essere lievemente migliorate:
abbiamo fatto variare i δ in modo tale che $\bigcup_{i \in \mathbb{N}} [2\delta, 3\delta]$ fosse una semiretta di $\mathbb{R}$.
Possiamo scegliere i δ in modo tale che ricoprano meno, ma contengano comunque tutti i valori interi maggiori di quello fissato.
Ad esempio nel Teorema sul numero di δ-run dobbiamo ricoprire tutti i periodi dal 10 in su. possiamo allora scegliere:

- $\delta_0 = \frac{10}{2}$ che ricopre i periodi da 10 a 15.
- $\delta_1 = \frac{16}{2}$ che ricopre da 16 a 24.

E cosi' via, ma il risparmio che si ottiene non raggiunge gli $0.05n$.

Conclusioni personali

La svolta nell'algoritmica delle ripetizioni si è avuta col passaggio dalla ripetizione, alla struttura relativamente piú complessa del run: Esso riassume un gruppo di ripetizioni successive.
Ci chiediamo: possiamo trovare strutture piú sintetiche dei run, ossia che ci permettano un risparmio di memoria e di tempo e da cui poi sia eventualmente possibile ricavare tutti i run?
Dalla dimostrazione del Teorema sul numero di δ-run vediamo come due run vicini diano origine ad una stringa molto regolare ma molto ricca di run, composta da due run successivi, sfasati, con la stessa stringa generatrice.
Raggruppiamo allora tutti i run successivi relativi alla stessa stringa generatrice in una struttura massimale.
D'ora in avanti chiamiamo questa struttura *multi-run*.

Conclusioni personali

La svolta nell'algoritmica delle ripetizioni si è avuta col passaggio dalla ripetizione, alla struttura relativamente piú complessa del run: Esso riassume un gruppo di ripetizioni successive.
Ci chiediamo: possiamo trovare strutture piú sintetiche dei run, ossia che ci permettano un risparmio di memoria e di tempo e da cui poi sia eventualmente possibile ricavare tutti i run?
Dalla dimostrazione del Teorema sul numero di δ-run vediamo come due run vicini diano origine ad una stringa molto regolare ma molto ricca di run, composta da due run successivi, sfasati, con la stessa stringa generatrice.
Raggruppiamo allora tutti i run successivi relativi alla stessa stringa generatrice in una struttura massimale.
D'ora in avanti chiamiamo questa struttura *multi-run*.

possibile utilizzo

Per semplicità chiamiamo *primordiale* una stringa x che non è del tipo y^β per ogni y stringa e $\beta \geq 2$ reale.
Ad esempio *ababa*, *ababa* è un run di esponente 2, e la sua generatrice *ababa* è un run di esponente 2.5, ossia non è primordiale.
Un primo utilizzo consiste nel memorizzare tutti i multi-run con generatrice primordiale, e tutti gli altri multi-run a patto che abbiano almeno due run.
In formule: data una qualsiasi x non primitiva, salvo i multi-run del tipo $x^{\alpha_1}, \ldots, x^{\alpha_t}, x^{\alpha_{t+1}}$ con $t \geq 2$ e $\alpha_{t+1} \geq 0$, mentre scarto quelli del tipo $x^{\alpha_1}, x^{\alpha_2}$ con $0 \leqslant \alpha_2 < 2$.
Il risparmio consiste in tutti i run non primordiali, non succeduti da run con la stessa generatrice.
Un secondo approccio alternativo porta a risparmiare tutti i run con generatrice una ripetizione, che a sua volta abbia una generatrice primordiale.

possibile utilizzo

Per semplicità chiamiamo *primordiale* una stringa x che non è del tipo y^β per ogni y stringa e $\beta \geq 2$ reale.
Ad esempio *ababa*, *ababa* è un run di esponente 2, e la sua generatrice *ababa* è un run di esponente 2.5, ossia non è primordiale.
Un primo utilizzo consiste nel memorizzare tutti i multi-run con generatrice primordiale, e tutti gli altri multi-run a patto che abbiano almeno due run.
In formule: data una qualsiasi x non primitiva, salvo i multi-run del tipo $x^{\alpha_1}, \ldots, x^{\alpha_t}, x^{\alpha_{t+1}}$ con $t \geq 2$ e $\alpha_{t+1} \geq 0$, mentre scarto quelli del tipo $x^{\alpha_1}, x^{\alpha_2}$ con $0 \leqslant \alpha_2 < 2$.
Il risparmio consiste in tutti i run non primordiali, non succeduti da run con la stessa generatrice.
Un secondo approccio alternativo porta a risparmiare tutti i run con generatrice una ripetizione, che a sua volta abbia una generatrice primordiale.

possibile utilizzo

Per semplicità chiamiamo *primordiale* una stringa x che non è del tipo y^β per ogni y stringa e $\beta \geq 2$ reale.
Ad esempio *ababa*, *ababa* è un run di esponente 2, e la sua generatrice *ababa* è un run di esponente 2.5, ossia non è primordiale.
Un primo utilizzo consiste nel memorizzare tutti i multi-run con generatrice primordiale, e tutti gli altri multi-run a patto che abbiano almeno due run.
In formule: data una qualsiasi x non primitiva, salvo i multi-run del tipo $x^{\alpha_1}, \ldots, x^{\alpha_t}, x^{\alpha_{t+1}}$ con $t \geq 2$ e $\alpha_{t+1} \geq 0$, mentre scarto quelli del tipo $x^{\alpha_1}, x^{\alpha_2}$ con $0 \leqslant \alpha_2 < 2$.
Il risparmio consiste in tutti i run non primordiali, non succeduti da run con la stessa generatrice.
Un secondo approccio alternativo porta a risparmiare tutti i run con generatrice una ripetizione, che a sua volta abbia una generatrice primordiale.

ritrovamento dei run

Come ricaviamo tutti i run impliciti da un multi-run?
Il multi-run puó essere riassunto con la successione dei suoi esponenti $\alpha_1, \ldots, \alpha_{t-1}, \alpha_t$. Dato un $j \geq 1$, per ogni k naturale tale che $2 \leq \alpha_{j-1} - k \leq \alpha_j$ abbiamo un run di generatrice $x^{\alpha_{j-1}-k}$ al'interno del multi-run, e tutti e soli i run non memorizzati vengono ritrovati in questo modo.
La difficoltà nell'applicazione di questi metodi consiste nel riconoscimento delle stringhe primordiali.
I vantaggi sono una riduzione della dimensione dell'output.
Un algoritmo che riuscisse a trovare i multi-run senza dover trovare i run avrebbe dei grossi vantaggi.

ritrovamento dei run

Come ricaviamo tutti i run impliciti da un multi-run?
Il multi-run puó essere riassunto con la successione dei suoi esponenti $\alpha_1, \ldots, \alpha_{t-1}, \alpha_t$. Dato un $j \geq 1$, per ogni k naturale tale che $2 \leq \alpha_{j-1} - k \leq \alpha_j$ abbiamo un run di generatrice $x^{\alpha_{j-1}-k}$ al'interno del multi-run, e tutti e soli i run non memorizzati vengono ritrovati in questo modo.
La difficoltà nell'applicazione di questi metodi consiste nel riconoscimento delle stringhe primordiali.
I vantaggi sono una riduzione della dimensione dell'output.
Un algoritmo che riuscisse a trovare i multi-run senza dover trovare i run avrebbe dei grossi vantaggi.