Cenni alla costruzione di un SuffixTree

Lezione 5

Giuditta Franco

Dipartimento di Informatica,
Università di Verona

19 Febbraio 2008

Talk Outline

1. Definizione di Suffix Tree
2. Applicazioni
3. Come costruire un Suffix Tree

Un esempio introduttivo

L'albero dei suffissi di una data stringa è una struttura dati che consente di memorizzare tutti i suffissi della stringa.

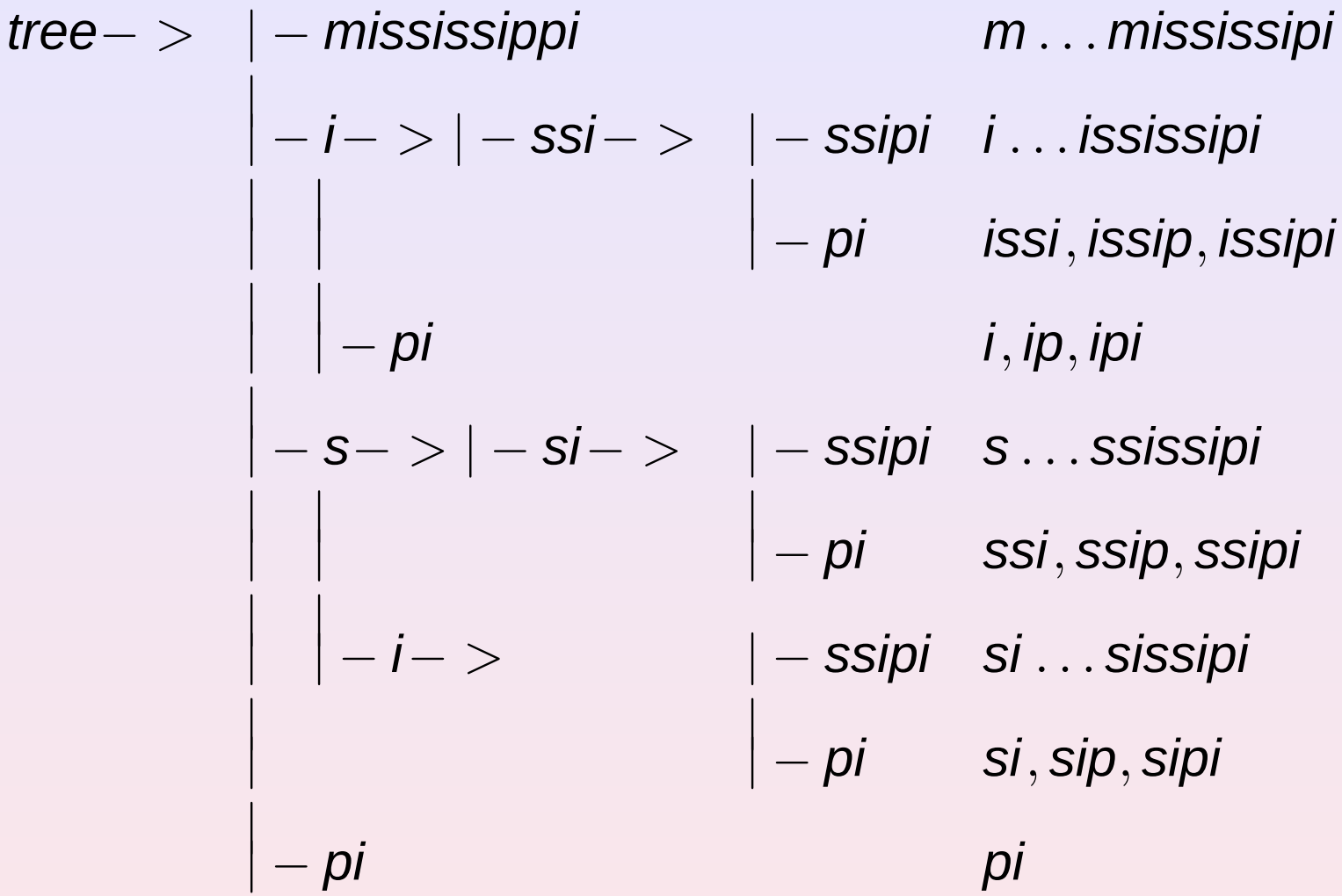
Se T è la stringa $t_1 t_2 ... t_i ... t_n$, allora $T_i = t_i t_{i+1} ... t_n$ è il suffisso di T che parte dalla posizione i, denotato anche dalla coppia di interi (i, n).

Per esempio, suffissi della stringa *mississipi* sono:
$T_1 = mississipi$, $T_2 = ississipi$, $T_3 = ssissipi$,
$T_4 = sissipi$, $T_5 = issipi$, $T_6 = ssipi$, $T_7 = sipi$, $T_8 = ipi$,
$T_9 = pi$, $T_{10} = i$, $T_{11} = (empty)$.

Albero dei suffissi della stringa *mississipi* (I)

- Si ordinino i suffissi lessicograficamente, e si osservi che hanno dei prefissi comuni.
- $$\begin{aligned} T_{11} &= \\ T_{10} &= i \\ T_{8} &= ipi \\ T_{5} &= issipi \\ T_{2} &= ississipi \\ T_{1} &= mississipi \\ T_{9} &= pi \\ T_{7} &= sipi \\ T_{4} &= sissipi \\ T_{6} &= ssipi \\ T_{3} &= ssissipi \end{aligned}$$

Albero dei suffissi della stringa *mississipi* (II)



Visualizzare esempi a volontà su
`www.allisons.org/ll/AlgDS/Tree/Suffix/`

Definizione di Albero dei Suffissi

Sia data una stringa S di lunghezza m.

- È un albero ordinato, direzionato, e radicato, con esattamente m foglie (numerate da 0 a $m-1$);
- ogni nodo interno (diverso dalla radice) ha almeno due figli, e ogni arco è etichettato con una sottostringa non vuota di S;
- le etichette su due archi uscenti da un nodo devono avere caratteri iniziali diversi;
- per ogni foglia i, la concatenazione delle etichette sugli archi del cammino dalla radice alla foglia i corrisponde al suffisso $S(i, m-1)$ di S.

Esistenza dell'Albero dei Suffissi

La definizione data non garantisce che esiste un suffix tree per ogni striga!

In particolare, tutte le volte che un suffisso della stringa si trova come prefisso di un altro suffisso, l'albero dei suffissi non esiste, perché il cammino per il primo suffisso non finirebbe in una foglia dell'albero.

Esempi: remore, xabxa, GATTAACTA!!

Soluzione: basta aggiungere un simbolo speciale che indichi la fine della stringa, di solito si usa il $.

Esistenza dell'Albero dei Suffissi

La definizione data non garantisce che esiste un suffix tree per ogni striga!

In particolare, tutte le volte che un suffisso della stringa si trova come prefisso di un altro suffisso, l'albero dei suffissi non esiste, perché il cammino per il primo suffisso non finirebbe in una foglia dell'albero.

Esempi: remore, xabxa, GATTAACTA!!

Soluzione: basta aggiungere un simbolo speciale che indichi la fine della stringa, di solito si usa il $.

Applicazioni

L'albero dei suffissi permette di risolvere velocemente tanti problemi su (un insieme dato di) stringhe. Sono utilizzati per:

- la ricerca di una una sottostringa, in tempo lineare (non è banale, considerando che il numero di sottostringhe è quadratico con la lunghezza della stringa), e.g., ricerca della più lunga sottostringa che appare due volte in una sequenza biologica, in un testo cinese;

- compilatori, analizzatori sintattici, completamento dei comandi di shell, gli indirizzi del web, di posta elettronica, gli sms!

Un storia breve

Il primo algoritmo di costruzione di un suffix tree in tempo lineare fu dato da Weiner nel 1973.

Tre anni dopo McCreight ha scritto un algoritmo che processava i testi da destra a sinistra, in tempo lineare, ma con un'occupazione di spazio più efficiente.

Solo negli anni 1992-1995 Ukkonen ha dato un algoritmo che processa il testo da sinistra a destra, mantenendo il tempo lineare e tutti i vantaggi dell'algoritmo di McCreight. Questo algoritmo mantiene un suffix tree del prefisso i-esimo della stringa, ad ogni passo i, per i che varia da 1 a n (lunghezza della stringa).

Algoritmo di costruzione (naive)

Sia T una stringa di lunghezza n. Algoritmo ricorsivo, per accrescimento dell'albero N_i di tutti i suffissi da 1 a i. In particolare N_1 ha due nodi (radice con elemento T, e foglia numerata 0) e l'arco con etichetta \$.

N_{i+1}, albero dei suffissi $T[i+1, n]$, si ottiene nel modo seguente:

1. Si trovi il cammino più lungo etichettato con un prefisso di $S[i+1, n]$.

2. Si consideri il "matching" di S[i+1,n] con le etichette del cammino concatenate, e si inserisca un nuovo nodo w appena finisce il "matching", creando un nuovo arco (w,i+1) con il suffisso di S[i+1,n] che non coincide con le etichette del cammino.

Commenti

1. Il cammino trovato è unico e finisce in un nodo interno, per le proprietà dell'albero: da un nodo non possono uscire etichette con la stessa iniziale e T non ha suffissi che sono prefissi di altri suffissi.

2. Così facendo si mantengono le proprietà dell'albero.

Ciascuno dei nodi richiede l'allocazione di un array di dimensione uguale all'alfabeto (per alfabeti grossi sorge un problema di spazio).

Questo algoritmo ha complessità quadratica con la lunghezza della stringa.

Per approfondimenti sulle implementazioni con java, vedere
`http://ai-agents.com/Javadoc/AIAgents/ \`
`SuffixTree/Java/suffixTree.html`