Arbres des suffixes

Suffix trees

Sèverine Bérard

novembre 2020

ISE-M – FDS, Université de Montpellier

Plan du cours

Plan du cours

Arbres des suffixes

- Structure de données arborescente contenant tous les suffixes d'un texte (T un texte de longueur n)

- Utilisé pour l'indexation des textes et la recherche exacte de motif (P de taille m)

Avec les meilleurs algorithmes comme Ukkonen

$\longrightarrow$ Construction de l'arbre pour un texte T en $O(n)$
$\longrightarrow$ Recherche de P en $O(m)$
$\longrightarrow$ Stockage de T en $O(n)$

Historique

- 1973 : Premier algorithme linéaire par Weiner : *position tree*
- 1976 : McCreight propose une amélioration de l'espace mémoire utilisé
- 1995 : Ukkonen propose un algorithme linéaire conceptuellement différent des deux premiers mais gardant tous les avantages temps et mémoire et plus simple à expliquer

"The algorithm of 73" d'après Knuth

Cette structure de donnée permet de résoudre de très nombreux problèmes de façon très efficace, par exemple chercher

- un motif avec un nombre d'erreur fixé
- la plus longue sous-chaîne commune
- les plus longues sous-chaînes répétées
- des palindromes maximaux
- ...

Plan du cours

Arbre des suffixes (alphabet fini et connu)

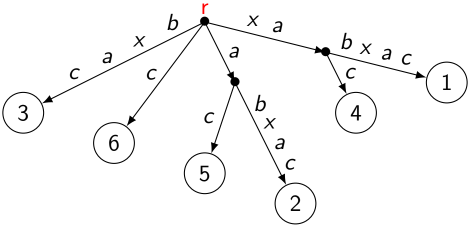
Un *arbre des suffixe* $\mathcal{T}$ pour une séquence S de longueur n est une arborescence (arbre orienté enraciné) :

1. ayant exactement n feuilles numérotées de 1 à n
2. chaque nœud interne ($\neq$ racine) a au moins 2 fils
3. chaque arête est étiquetée avec une sous-chaîne non vide de S
4. deux arcs sortant d'un même nœud ne peuvent pas commencer par la même lettre

Caractéristique principale

Pour chaque feuille i (de 1 à n), la concaténation des étiquettes des arcs de la racine jusqu'à i est exactement le suffixe de S qui commence à la position i, c.-à-d. $S[i..n]$

Exemple $S = xabxac$



Attention

La définition donnée ne garantie pas l'existence d'un arbre des suffixes pour toute chaîne S. Ex : si un des suffixes de S est préfixe d'un autre suffixe.

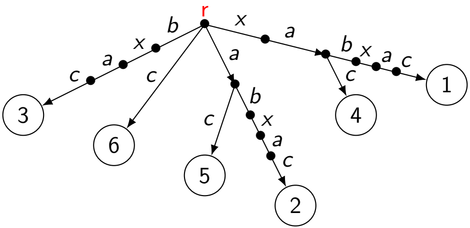
$\Rightarrow$ Caractère sentinelle, généralement \$

Exemple : essayez de construire l'arbre des suffixes pour S=`tata`, vous n'arriverez pas à avoir les feuilles numérotées 3 avec comme étiquette-chemin `ta`, et 1 avec comme étiquette-chemin `tata` sans violer une des contraintes de la définition.
Par contre, vous pourrez construire l'arbre des suffixes de `tata$` sans difficulté. Il a 5 feuilles et l'étiquette-chemin de la feuille numérotée 5 est `$`.

Relation avec le dictionnaire vu dans Aho-Corasick

- Ensemble $\mathcal{P}$ des motifs = ensemble des suffixes de S
- On peut obtenir l'arbre des suffixes de S en fusionnant les chemins sans embranchement en un seul arc

$\Rightarrow$ en utilisant l'algo de AC on peut donc construire un arbre des suffixes en $O(n^2)$



Dictionnaire pour $\mathcal{P} = \{xabxac, abxac, bxac, xac, ac, c\}$

Quelques définitions

Étiquette-chemin

L'étiquette d'un chemin de la racine à un nœud est la concaténation, dans l'ordre, des étiquettes des arcs de ce chemin

L'*étiquette-chemin* d'un nœud est l'étiquette du chemin de la racine à ce nœud

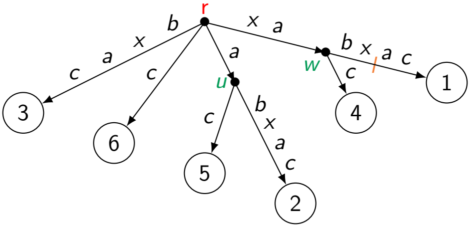
Profondeur de chaîne

Pour tout nœud v d'un arbre des suffixes, la profondeur de chaîne de v est le nombre de caractères dans son étiquette-chemin

Étiquette d'un chemin se terminant au milieu d'un arc

Un chemin qui finit au milieu d'un arc (u, v) coupe l'étiquette de (u, v) à ce point. L'étiquette d'un tel chemin est l'étiquette-chemin de u concaténée avec les caractères de (u, v) jusqu'au point de coupure

Retour sur l'exemple $S = xabxac$



- L'étiquette chemin de w est xa
- La chaîne a étiquette un chemin qui finit au nœud u
- La chaîne $xabx$ étiquette un chemin qui finit à l'intérieur de l'arc $(w, 1)$

Plan du cours

Problème de recherche exacte

Trouver toutes les occurrences d'un motif P de longueur m à l'intérieur d'un texte T de longueur n

Idée clé

Chaque occurrence de P est préfixe d'un suffixe de T

Approche avec un arbre des suffixes

1. Construire $\mathcal{T}$ l'arbre des suffixes de T $O(n)$
2. Mettre en correspondance les caractères de P le long de l'unique chemin de $\mathcal{T}$ jusqu'à :
   a) ce que P soit complètement épuisé
   b) ou plus de correspondance possible

Résultats :

b) P n'apparaît pas dans T $O(m)$

a) chaque feuille dans le sous-arbre raciné au dernier match est numérotée avec une position de départ de P dans T et chaque position de départ de P dans T numérote une telle feuille

   $\rightarrow$ collecter les k positions de P en parcourant linéairement le sous-arbre et noter tous les numéros de feuilles
   (nb feuilles prop. nb d'arcs car au moins 2 fils/nœud $\Rightarrow O(k)$)

$O(m+k)$

Plan du cours

Algorithme : Approche naïve

Données : Le texte T de longueur n

$\mathcal{T} := ArbreVide$;

pour *(i de 1 à n)* **faire**

  Insérer l'arc $T[i..n]\$$ dans $\mathcal{T}$;

Soit N_i l'arbre qui encode tous les suffixes de 1 à i, pour passer de N_i à N_{i+1} :

1. Partir de la racine de N_i
2. Trouver le plus long chemin depuis la racine qui correspond à un préfixe de $T[i+1..n]\$$
   $\rightarrow$ *un seul chemin possible car aucun suffixe de $T\$$ n'est préfixe d'un autre suffixe, on est alors soit à un nœud w ($\neq$ feuille) ou au milieu d'une arête*
3. Insérer éventuellement un nouveau nœud et créer un nouvel arc
   - étiqueté avec les derniers caractères de $T[i+1..n]\$$ non mis en correspondance
   - finissant sur une nouvelle feuille étiquetée $i+1$

Procédure en $O(n^2)$

Plan du cours

Références

Toute cette présentation est basée sur la section 3.4 du livre suivant :

[Gusfield, 97] Dan Gusfield, **Algorithms on Strings, Trees and Sequences** - Computer Science and Computational Biology, University of California, Davis. ISBN :9780521585194. Août 1997. *En anglais*