Tutoriat 4

Arbori de Sufixe Generalizați

1. Introducere

În acest tutoriat, vom explora ce este un arbore de sufixe generalizat și cum poate fi folosit, de exemplu, pentru găsirea celui mai lung subșir comun. Găsirea celui mai lung subșir comun reprezintă un caz particular al metodelor de similaritate a șirurilor, în care se caută subsecvențe comune – iar în acest context, „secvența” se referă la subșiruri.

Vom ilustra o aplicație a arborelui de sufixe generalizat prin rezolvarea problemei de identificare a celui mai lung subșir comun. Mai apoi, vom construi un arbore de sufixe generalizat, pornind de la construirea unui trie de sufixe și a unui trie Patricia (folosind terminologia specifică domeniului), apoi adnotând acești arbori pentru a obține structura finală.

2. Trie de Sufixe

Pentru a răspunde la întrebarea inițială, nu, nu am greșit cuvântul „arbore”. În limbajul arborilor de sufixe, un **trie** reprezintă o structură intermediară în construirea unui arbore de sufixe generalizat complet, util pentru diverse sarcini. Un trie de sufixe este un arbore în care muchiile (adică liniile care leagă nodurile) sunt etichetate cu literele din sufixele unui șir.

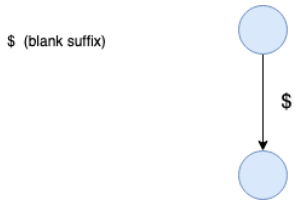
Construim un arbore de sufixe parcurgând fiecare sufix și creând o muchie pentru fiecare caracter, începând de la nodul rădăcină. Dacă sufixul care trebuie inserat începe cu o secvență de caractere ce există deja în arbore, se urmează acele caractere până la prima diferență, moment în care se creează o nouă ramură.

Acest concept este cel mai bine explicat printr-un exemplu. Să luăm cuvântul **nonsense**. Acest cuvânt are 8 sufixe, plus un sufix vid (notat cu `$`):

1. `$`
2. `e`
3. `se`
4. `nse`
5. `ense`
6. `sense`

7. nsense
8. onsense
9. nonsense

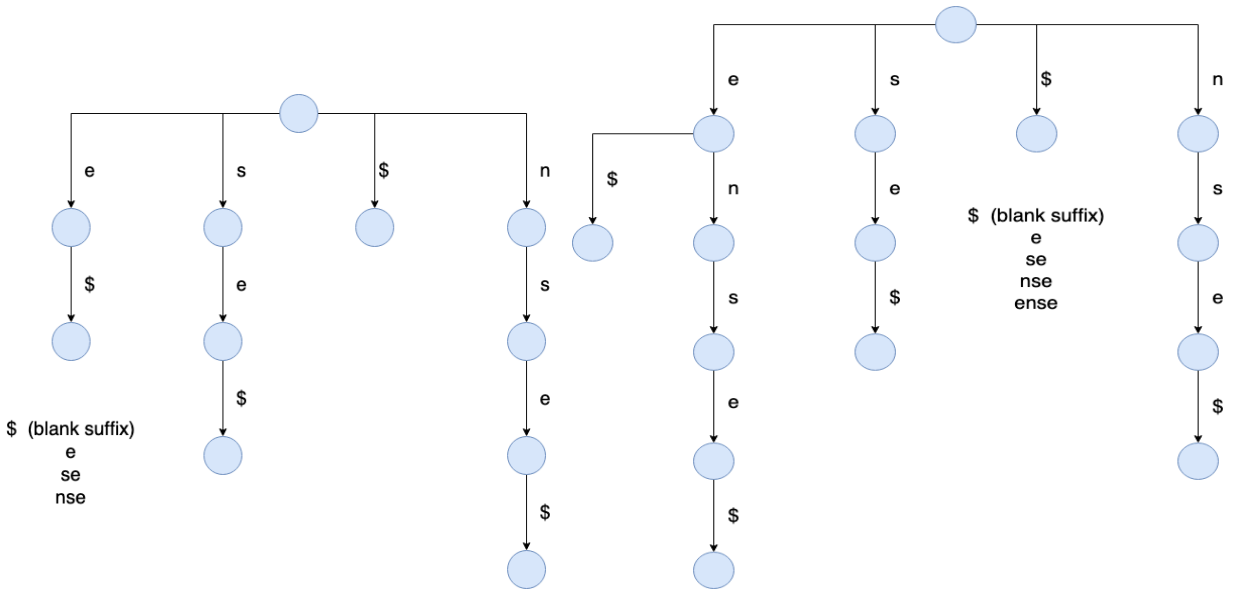
Începem construirea trie-ului de sufixe cu nodul de pornire și sufixul vid, etichetat cu $. Ulterior, conectăm la nodul de pornire prima literă a fiecărui sufix. Observăm că primele trei sufixe non-goale încep cu litere diferite, așa că se adaugă trei ramuri distincte.

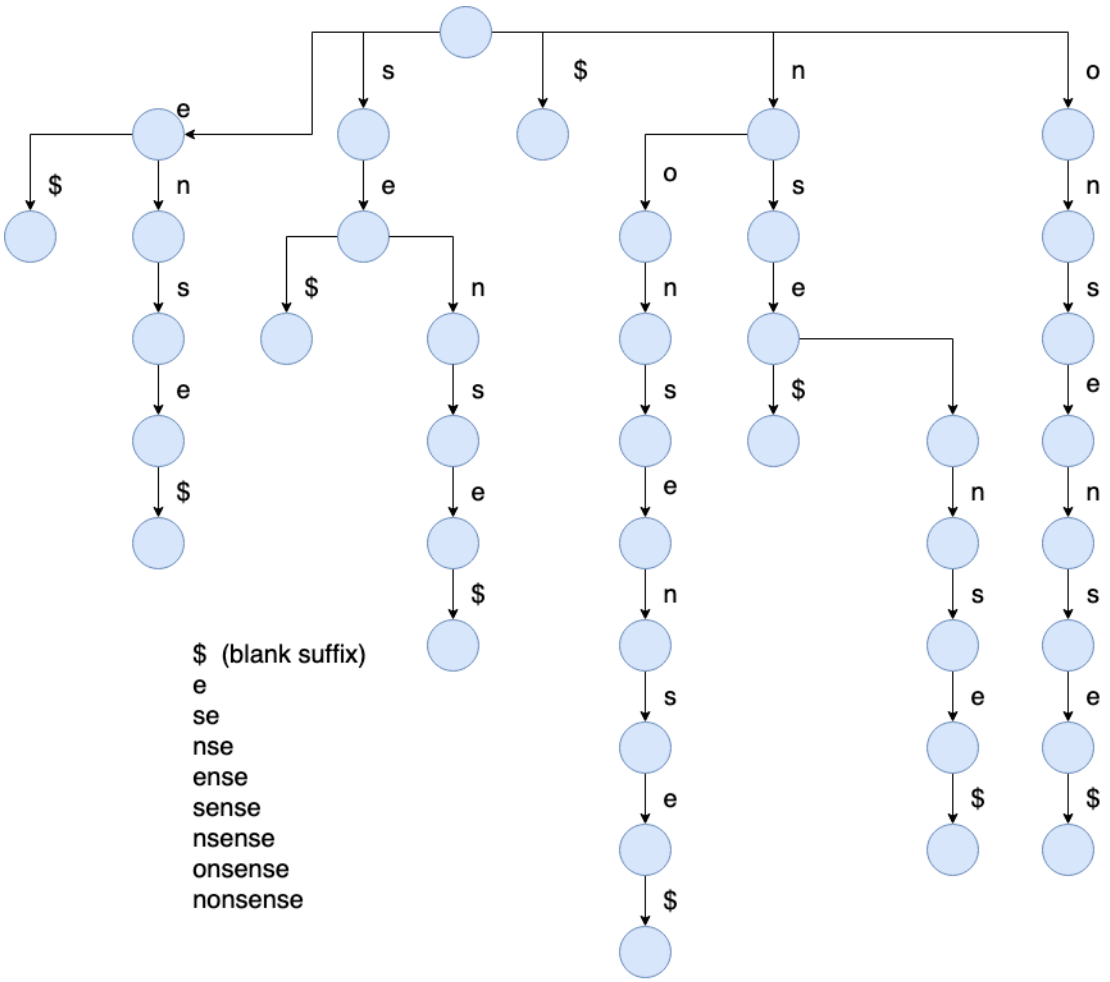


Următorul sufix, **ense**, începe cu e, care deja este prezent ca nod de la rădăcină. Se adaugă o ramură suplimentară la nodul e, urmând secvența comună (în acest caz, doar e) și apoi ramificând unde apar diferențe.

Continuăm acest proces, adăugând ramuri suplimentare ori de câte ori sufixele diferă, până când se finalizează construcția arborelui. La final, obținem un arbore de sufixe în care, de exemplu, se poate observa că există trei sufixe care încep cu n și că secvențele comune urmează aceeași ramură (precum nse pentru sufixele nse și nsense).

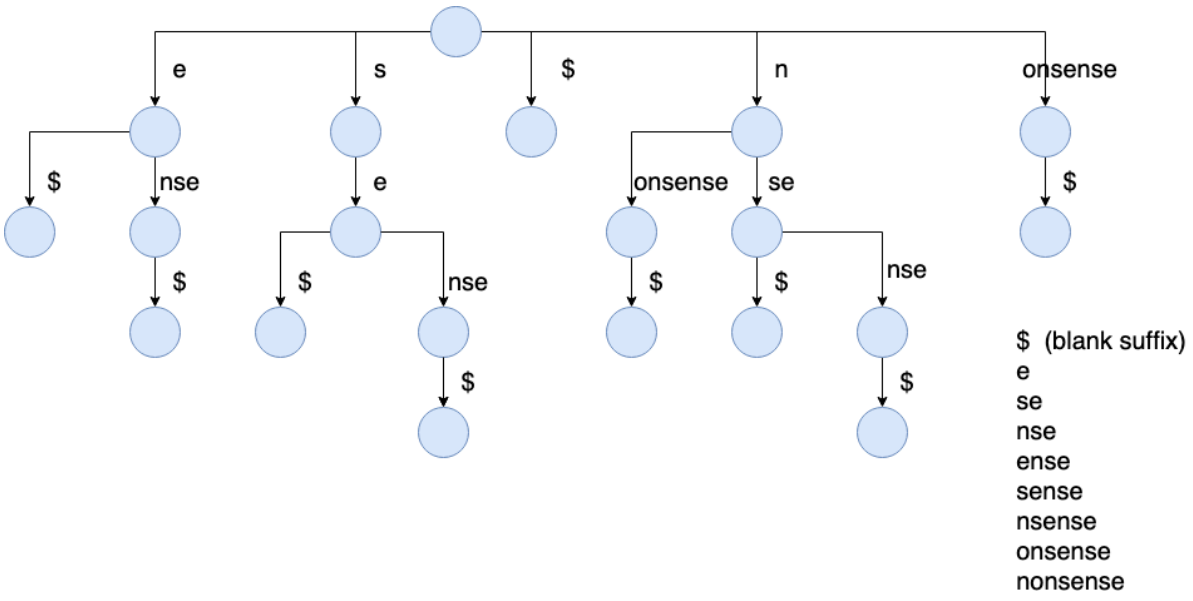
De asemenea, se constată că, dacă dimensiunea șirului T este m (adică |T| = m), arborele de sufixe are exact m + 1 noduri frunză. În cazul cuvântului **nonsense**, arborele construit are 9 noduri frunză. Complexitatea algoritmului depinde de numărul de noduri, motiv pentru care apare întrebarea:





Putem reduce numărul de noduri din trie?

Răspunsul, explicat în secțiunea următoare, constă în construirea unui **trie Patricia**.



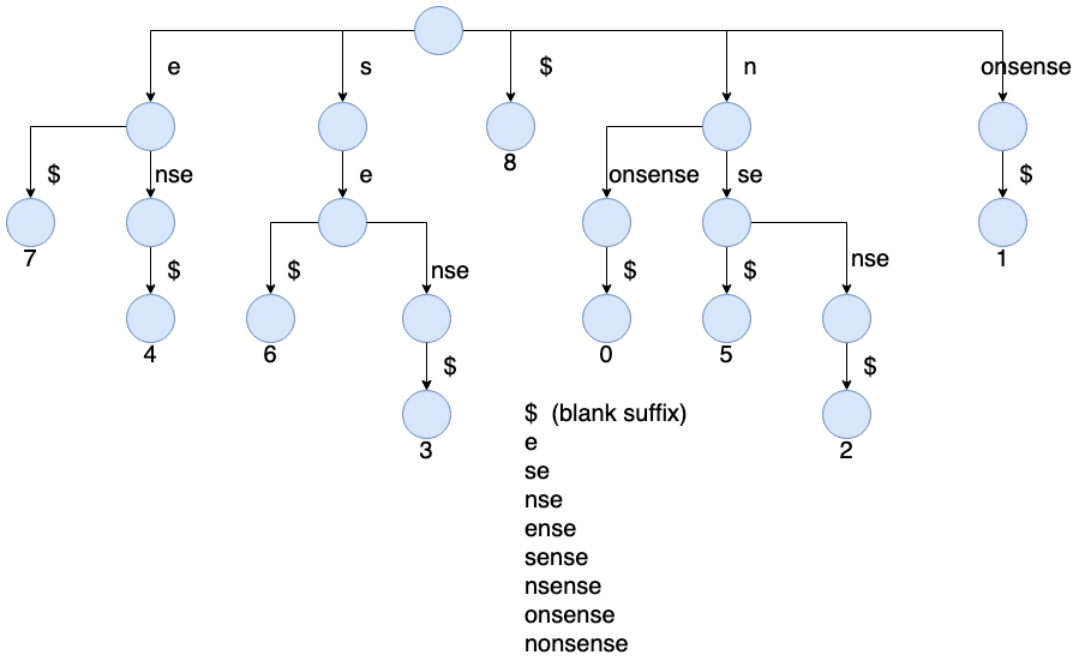
3. Patricia Trie

Un **trie Patricia** este, practic, arborele de sufixe în care toate nodurile „simple” (cele care au doar un singur copil și nu generează ramificații) sunt comprimate într-un singur nod. Folosind exemplul anterior, obținem o reducere a numărului de noduri, însă numărul de noduri frunză rămâne egal cu numărul de sufixe.

Construirea trie-ului Patricia reprezintă un pas intermediar esențial în obținerea arborelui de sufixe (și, ulterior, a arborelui de sufixe generalizat) folosit pentru diverse aplicații de recunoaștere a subșirurilor.

4. Arborele de Sufixe

Odată cu obținerea trie-ului Patricia, suntem cu un pas mai aproape de a construi o structură de date eficientă pentru recunoașterea subșirurilor. Un **arbore de sufixe** pentru un șir T este, în esență, un trie Patricia pentru T, în care fiecare nod frunză este etichetat cu indicele de start al sufixului corespunzător din T. Această etichetă ne oferă o legătură directă la sufixul original, facilitând astfel diverse operații de căutare și analiză.

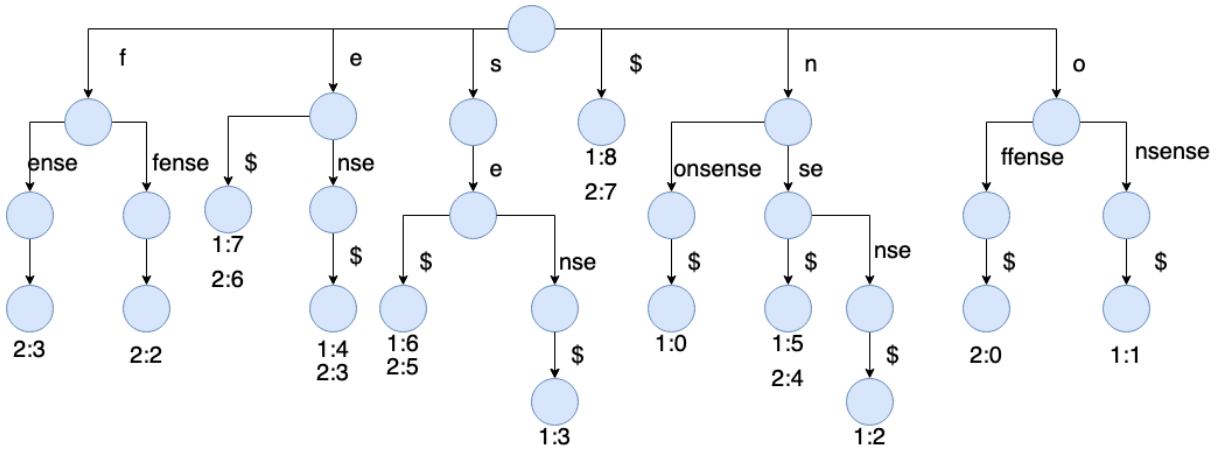


5. Arborele de Sufixe Generalizat

Un arbore de sufixe clasic descrie doar un singur șir T. Totuși, versiunea **generalizată** a acestei structuri poate indexa simultan mai multe șiruri. În acest caz, rezultatul unei operații de căutare poate indica, de exemplu, care dintre șiruri conțin un anumit subșir.

Un **arbore de sufixe generalizat** pentru șirurile T_1, T_2, …, T□ este un arbore de sufixe construit pe toate aceste șiruri, dar etichetele nodurilor frunză includ nu doar poziția de start în șir, ci și un indice care identifică șirul din care face parte sufixul (notat, de exemplu, ca *i:j*, unde *j* este poziția sufixului în șirul T_i).

Vom ilustra acest concept cu un exemplu ce implică două șiruri: **T_1 = nonsense** și **T_2 = offense**.



8 $ (blank suffix)
7 e
6 se
5 nse
4 ense
3 sense
2 nsense
1 onsense
0 nonsense

7 $ (blank suffix)
6 e
5 se
4 nse
3 ense
2 fense
1 ffense
0 offense

6. Cel Mai Lung Subșir Comun

Arborele de sufixe generalizat poate fi folosit eficient pentru recunoașterea subșirurilor. Ideea de bază este că orice subșir dintr-un șir este, de fapt, un prefix al unui sufix al acelui șir. Cu alte cuvinte, inserând toate sufixele unui șir în arbore, introducem și toate posibilele prefixe (deoarece fiecare prefix este începutul unui sufix). Astfel, două subșiruri pot fi comparate prin parcurgerea comună a arborelui.

Algoritmul pentru găsirea celui mai lung subșir comun se desfășoară în trei pași:

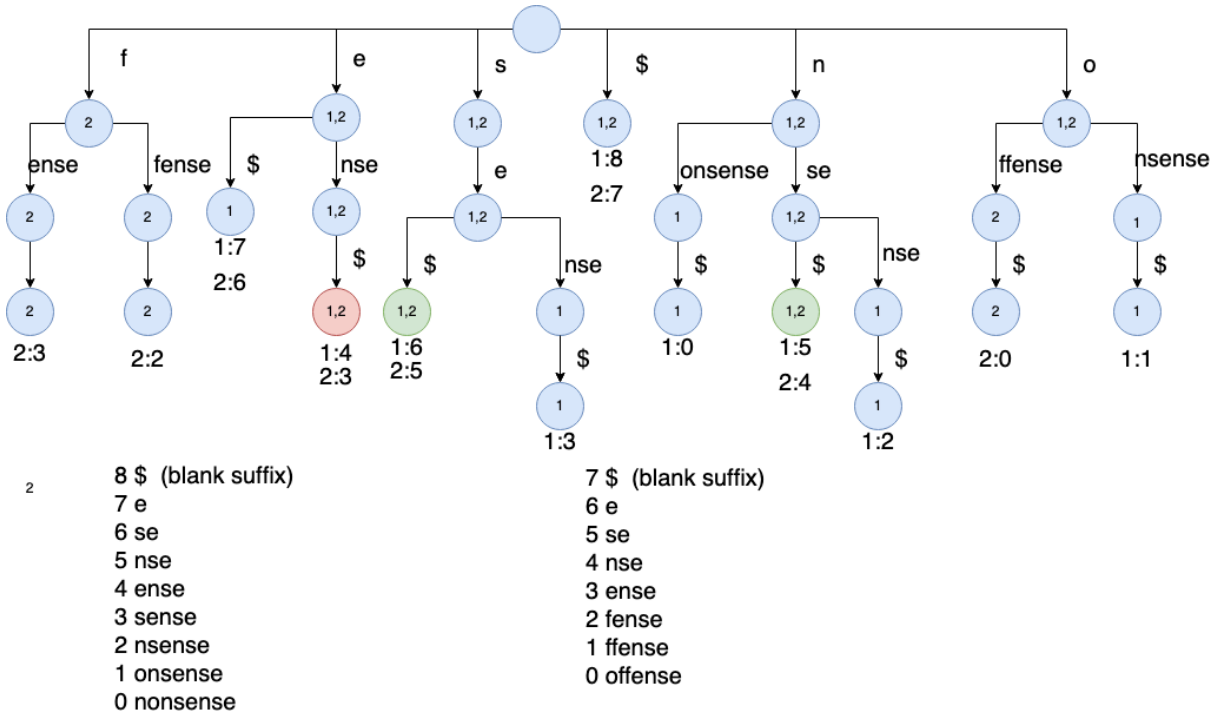
1. **Construiește arborele de sufixe generalizat** pentru T_1 și T_2.

2. **Adnotează fiecare nod intern** din arbore cu informația dacă acesta conține cel puțin un nod frunză din fiecare șir (adică dacă subșirul respectiv apare atât în T_1, cât și în T_2).

3. **Execută o căutare în adâncime** (depth-first search) în arbore pentru a identifica nodul marcat care are cea mai mare adâncime a subșirului.

În exemplul nostru, după ce am construit arborele pentru șirurile **nonsense** și **offense** și am adnotat nodurile conform prezenței șirurilor, observăm că nodul evidențiat (de exemplu, colorat în roșu) corespunde subșirului **ense**, care reprezintă cel mai lung subșir comun.

Nodurile de culoare verde, reprezentând subșiruri mai scurte precum **nse** și **se**, se află la același nivel în arbore, datorită compresiei realizate în trie-ul Patricia.



7. Concluzie

Acest articol a prezentat modul de construire a unui arbore de sufixe generalizat pentru rezolvarea unei probleme de recunoaștere a subșirurilor. Metoda expusă este una simplă și intuitivă pentru generarea arborilor de sufixe generalizați, însă există și multe tehnici avansate pentru optimizarea algoritmului – cum ar fi construcția online a arborilor de sufixe, dezvoltată la Universitatea din Helsinki.

De asemenea, există numeroase surse de cod disponibile pentru implementarea acestei structuri în diverse limbaje de programare, cum ar fi C++, Python sau Java.

8. Referinte:

https://www.cs.yale.edu/homes/aspnes/pinewiki/SuffixArrays.html

https://www.baeldung.com/cs/generalized-suffix-trees