Алгоритм синтаксичного аналізу Кока–Касамі–Янгера для граматик у нормальній формі Хомського

Існує достатньо ефективний метод визначення чи належить ланцюжок мові, що задана граматикою, який оснований на ідеї "динамічного програмування". Цей алгоритм відомий ще як СҮК-алгоритм — алгоритм Кока—Янгера—Касамі.

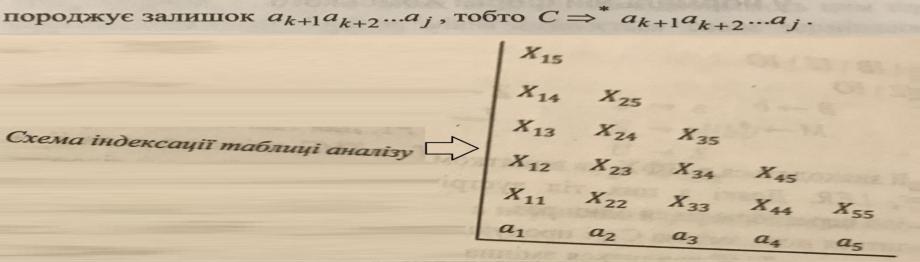
Він використовується лише для граматик у нормальній формі Хомського. На вхід алгоритму подається ланцюжок $w = a_1 a_2 \dots a_n$ з T^* .

За час $O(n^3)$ алгоритм будує таблицю, яка говорить, чи належить w мові L.

Алгоритм Кока-Касамі-Янгера для синтаксичного аналізу.

<u>Базис</u>. Обчислюємо перший рядок так. Оскільки ланцюжок, _{який} починається та закінчується в позиції і, являє собою просто термінал аі, а граматика знаходиться в НФХ, єдиний спосіб породити а_і полягає в використаннні продукції вигляду $A \to a_i$ граматики G. Отже, X_{ii} є множиною змінних A, для яких $A \to a_{i-}$ продукція G.

Індукція. Нехай потрібно обчислити X_{ij} в (j-i+1)-му рядку, і всі множини X в нижніх рядках вже обчислені, тобто відомі для всіх подланцюжків, коротших, ніж $a_ia_{i+1}\dots a_j$, і зокрема, для всіх власних префіксів і суфіксів цього ланцюжка. Можна припустити, що j-i>0, оскільки випадок j=i розглянуто в базисі. Тому будь-який вивід $A\Rightarrow^*a_ia_{i+1}\dots a_j$ має починатися кроком $A\to BC$. Тоді В породжує деякий префікс рядка $a_ia_{i+1}\dots a_j$, скажімо, $B\Rightarrow^*a_ia_{i+1}\dots a_k$ для деякого k< j. Відповідно, C породжує залишок $a_{k+1}a_{k+2}\dots a_j$, тобто $C\Rightarrow^*a_{k+1}a_{k+2}\dots a_j$.



Алгоритм Кока-Касамі-Янгера для синтаксичного аналізу.

Доходимо висновку, що для того, щоб A потрапило в X_{ij} , потрібно знайти змінні B і C і ціле k, при яких справедливі такі умови:

- 1) $i \leq k \leq j$;
 - 2) B належить X_{ik} ;
 - 3) C належить $X_{k+1,j}$;
 - 4) $A \rightarrow BC$ продукція в G.

Пошук таких змінних А потребує обробки не більше п пар обчислених раніше множин: (X_{ii}, X_{i+1}, j) , $(X_{i}, i+1, X_{i+2}, j)$ і т. д. до $(X_{i}, j-1, X_{jj})$. Отже, ми піднімаємося по колонці, розташованій під X_{ij} , і одночасно спускаємося по діагоналі.

