4. Контекстно-свободни граматики и езици. Стекови автомати

Анотация: Контекстно-свободни граматики. Дървета за синтактичен анализ. Нормална форма на Чомски. Стекови автомати. Връзка между стековите автомати и контекстно-свободните граматики (доказателство в едната посока по избор). Свойства на затвореност. Лема за покачването (xyzuv) (с доказателство). Примери за езици, които не са контекстно-свободни.

Контекстно-свободни граматики

Контекстно-свободните граматики (КСГ) са генератори (или наричани още разпознаватели) на езици, които от дадена начална дума, позволяват краен брой ограничени правила за преобразувания.

Дефиниция (КСГ). Контекстно-свободна граматика наричаме четворката $G = \langle \Sigma, V, S, R \rangle$, където:

1) Σ е крайно множество от терминални символи (азбука на граматиката);
2) V е крайно множество от нетерминални символи (променливи) и $V \cap \Sigma = \emptyset$;
3) $S \in V$ е начален нетерминал;
4) $R \subseteq V \times (\Sigma \cup V)^*$ е крайно множество от правила, които преобразуват нетерминален символ в последователност от други символи.

Ще записваме терминалните символи с малки латински букви, нетерминалните с големи латински букви, думи от малки и големи букви ще бележим с малка гръцка буква, а правилата ще разделяме на лява и дясна част със стрелка надясно. Например: $a, b \in \Sigma$ са терминали; $S, T \in V$ са нетерминали; $\alpha, \beta \in (\Sigma \cup V)^*$ са думи, които могат да съдържат и терминали и нетерминали; $S \to aTb = \alpha$, $T \to bb = \beta$ са правила. Ще спазваме стриктно тази конвенция по-долу.

Правило 4) от дефиницията на КСГ издава от къде идва наименованието на тези генератори на езици. От лявата страна на всяко правило имаме единствен нетерминален символ, който се преобразува в последователност от други символи, без да се интересува какъв е контекста, който го заобикаля (останалите символи около него). Най-общия случай на граматика е когато позволим в лявата част на дадено правило отново да има израз от $(\Sigma \cap V)^*$ както в дясната. Тогава граматиките се наричат неограничени, тъй като нямаме никакви ограничения върху правилата.

Дефиниция (релацията $\Rightarrow$). За всеки две думи $\alpha_1, \alpha_2 \in (\Sigma \cup V)^*$ ще казваме, че са в релация $\alpha_1 \Rightarrow_G \alpha_2$, т.с.т.к. съществуват други две думи $\beta_1, \beta_2 \in (\Sigma \cup V)^*$ и нетерминал $T \in V$, за които е изпълнено $\alpha_1 = \beta_1 T \beta_2$, $\alpha_2 = \beta_1 \gamma \beta_2$ и $T \to \gamma$. Тоест от думата α_1 може да преобразуваме до думата α_2 чрез една стъпка (чрез прилагане веднъж на едно правило).

Когато граматиката се подразбира, може да пропускаме G в означението на релацията и да записваме накратко $\alpha_1 \Rightarrow \alpha_2$.

Дефиниция (релацията $\Rightarrow^*$). Релацията $\Rightarrow^*_G$ е рефлексивно и транзитивно затваряне на релацията $\Rightarrow_G$. Тоест, ако $\alpha_1 \Rightarrow^*_G \alpha_2$, то от думата α_1 може да отидем в думата α_2 чрез няколко (може и 0) стъпки.

Тази дефиниция е коректна и формално, но е твърде неописателна спрямо изчислителния модел, който може да описваме чрез граматики. Затова ще дадем следната еквивалентна дефиниция.

Дефинираме системата от правила:

$$\begin{cases} \dfrac{}{\alpha \overset{0}{\Rightarrow} \alpha} \ (0) & \text{за нула стъпки може да преобразуваме от думата } \alpha \text{ в думата } \alpha \\ & \text{над чертата няма нищо, тъй като твърдението е аксиома и не изисква предпоставки} \\ \dfrac{\alpha \to \beta \text{ и } \beta \overset{k}{\Rightarrow} \gamma}{\alpha \overset{k+1}{\Rightarrow} \gamma} \ (1) & \text{ако има правило } \alpha \to \beta \text{ и от } \beta \text{ може да} \\ & \text{преобразуваме до } \gamma \text{ за } k \text{ стъпки, то от } \alpha \text{ може да преобразуваме до } \gamma \text{ за } k+1 \text{ стъпки} \\ & \text{(това правило ни казва как може да удължим извода)} \\ \dfrac{\alpha_1 \overset{l_1}{\Rightarrow} \beta_1 \text{ и } \alpha_2 \overset{l_2}{\Rightarrow} \beta_2}{\alpha_1\alpha_2 \overset{l_1+l_2}{\Rightarrow} \beta_1\beta_2} \ (2) & \text{за } l_1 + l_2 \text{ стъпки може да преобразуваме от } \alpha_1\alpha_2 \text{ до } \beta_1\beta_2 \text{ при така} \\ & \text{зададените предпоставки} \end{cases}$$

Така дефинираната система е по-удачна, тъй като по понякога се налага да правим доказателства чрез индукция по дължината на извода или иначе казано по брой стъпки на изчислението и ни е по-удобно да имаме явния вид на броя стъпки на изчислението.

Сега може да дадем аналогична дефиниция на релацията $\alpha \Rightarrow^* \beta$ по следния начин: $(\exists l \in \mathbb{N}_0,\ l < \infty)\left[\alpha \overset{l}{\Rightarrow} \beta\right]$. Тоест вместо да дефинираме релацията чрез рефлексивно и транзитивно затваряне, казваме, че съществуват краен брой стъпки (или 0), които водят от думата α до думата β, чрез правила от граматиката.

Дефиниция (**език на граматика**). Езикът на граматиката G е $\mathscr{L}(G) = \{\omega \in \Sigma^* \,|\, S \Rightarrow^* \omega\}$ или аналогично $\mathscr{L}(G) = \{\omega \in \Sigma^* \,|\, S \overset{l}{\Rightarrow} \omega,$ за някое $l \in \mathbb{N}_0\}$.

Пример за граматика:

$$\Gamma = \langle \{a, b\}, \{S, T\}, S, \{S \to aSb \,|\, \varepsilon, aS \to bb\} \rangle.$$

Тази граматика има два нетерминала S и T, от който S е и начален. Азбуката на граматиката е $\{a, b\}$ и има 3 правила, две от които са обединени в едно чрез вертикална черта, поради еднаквата им лява страна. Забележете обаче, че така зададената граматика **НЕ Е** КСГ, тъй като правилото $aS \to bb$ зависи по някакъв начин от контекста на думата, за да може да се приложи. Въпреки това, Γ е валидна неограничена граматика.

Нека сега дадем пример за контекстно-свободна граматика (КСГ). Това е граматика, чиито правила имат лява страна само от един нетерминал.

Пример за КСГ:

$$\Gamma = \langle \{a, b\}, \{S\}, S, \{S \to aSb \,|\, \varepsilon\} \rangle$$

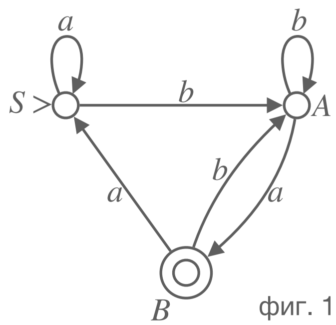
Нека $G = \langle \Sigma, V, S, R \rangle$ е КСГ, където $\Sigma = \{(,)\}$, $V = \{S\}$ и множеството от правила е $R = \{S \to \varepsilon, S \to SS, S \to (S)\}$. Така дефинираната граматика генерира всички правилно балансирани изрази от скоби. Това е така, защото всяка лява скоба може да бъде комбинирана с уникална следваща дясна скоба, а всяка дясна скоба може да бъде комбинирана с уникална предходна лява скоба. Освен това, думата между всеки две скоби от такава комбинация има същото свойство.

Две преобразувания по правилата на G са:

$$S \Rightarrow SS \Rightarrow S(S) \Rightarrow S((S)) \Rightarrow S(()) \Rightarrow (S)(()) \Rightarrow ()(())$$
$$S \Rightarrow SS \Rightarrow (S)S \Rightarrow ()S \Rightarrow ()(S) \Rightarrow ()((S)) \Rightarrow ()(())$$

Тоест една и съща дума може да се разпознава по повече от един път на преобразувания от КСГ.

Езикът, който задава граматиката G излиза от множеството на регулярните езици, тъй като съдържа всички думи от вида $(^*)^*$, които формират подезик $\tilde{L} = (^n)^n$, който лесно може да се докаже, чрез лемата за разширяването за регулярни езици, че не е регулярен език. Друг интересен факт е, че ако зададем граница $2n$ за максимален брой символи на всяка дума от G, то $|\mathscr{L}(G)| = \dfrac{1}{n+1}\dbinom{2n}{n}$ (числа на Каталан). От друга страна, всички регулярни езици са контекстно-свободни. Един лесен начин за доказателството на това твърдение е чрез директна конструкция на КСГ от автомата, които описва дадения регулярен език.
Нека например $\mathscr{A} = \langle Q, \Sigma, \delta, s, F \rangle$ е автомата, който разпознава даден регулярен език. Същия език се разпонзава от граматиката $G(\mathscr{A}) = \langle \Sigma, V, S, R \rangle$, където $V = Q$, $S = s$ и в R има два класа правила: $R = \{q \to ap : \delta(q, a) = p\} \cup \{q \to \varepsilon : q \in F\}$. Тоест нетерминалите са състоянията на автомата, а за всеки преход от q до p с буквата a имаме правило $q \to ap$.
Пример за построяване на КСГ по даден автоматен (регулярен) език:



фиг. 1

$$\Rightarrow \quad \begin{array}{l} S \to aS \\ S \to bA \\ A \to aB \\ A \to bA \\ B \to aS \\ B \to bA \\ B \to \varepsilon \end{array}$$

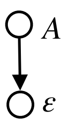
Дървета за синтактичен анализ
Дърветата за синтактичен анализ наричаме още дървета на извод на КСГ. Всяко едно такова дърво има върхове, на които съпоставяме буква от $\Sigma \cup V$. Деца ще имат само върхове с нетерминален етикет и тези деца ще се образуват като разбием израза на отделни букви от правилото за преобразувание на този нетерминал – като спазваме наредбата. Най-горния връх наричаме корен, а върховете на най-долното ниво от дадено разклонение наричаме листа. Всички листа са с етикети от терминален символ или празната буква. Извод на едно дърво на синтактичен анализ наричаме конкатенацията на всички негови листа от ляво на дясно.

Нека имаме произволна КСГ $G = \langle \Sigma, V, S, R \rangle$. Дефинираме нейното дърво на извод индуктивно:

1) 

фиг. 2

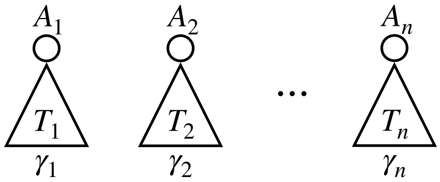
Това е дърво на извод за всяко $a \in \Sigma$. Единственият връх на това дърво на извод е едновременно и корен и листо. Изводът на това дърво е a.

2) Ако $A \to \varepsilon$ е правило от R, тогава 

фиг. 3

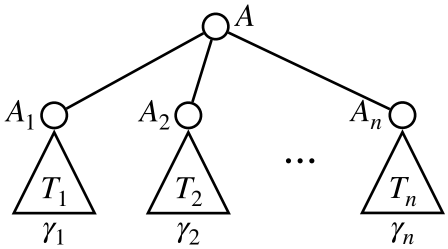
е дърво на извод. Неговият корен е с етикет на нетерминала A и неговото единствено листо е с етикет празната дума ε и извода на това дърво е ε.

3) Ако



фиг. 4

са дървета на извод, където $n \geq 1$, с корени с етикети $A_1, \ldots, A_n$ съответно и с изводи съответно $y_1, \ldots, y_n$ и $A \to A_1 \ldots A_n$ е правило в R, тогава



фиг. 5

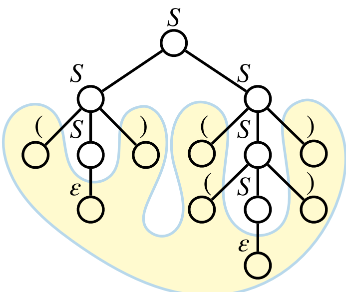
е дърво на извод. Неговият корен е новия връх с етикет нетерминала A, листата му са листата на неговите поддървета на извод и извода му е конкатенацията на изводите на поддърветата му на извод. Дърветата на извод обхващат различните начини при достигане до един и същ извод.

4) Нищо друго не е дърво за синтактичен анализ.

Дърветата на синтактичен анализ представлват еквивалентни класове за извод на дума от дадена граматика и подтискат несъществените разлики по отношение на реда на прилагане на правилата.

Да построим дърво на извод на двата еднакви извода, които разгледахме по-рано.

$$S \Rightarrow SS \Rightarrow S(S) \Rightarrow S((S)) \Rightarrow S(()) \Rightarrow (S)(()) \Rightarrow ()(())$$
$$S \Rightarrow SS \Rightarrow (S)S \Rightarrow ()S \Rightarrow ()(S) \Rightarrow ()((S)) \Rightarrow ()(())$$



фиг. 6

Може да окажем приоритет в граматиката, като просто подредим правилата ѝ в ред, в който да ги взимаме. Така ще може да построяваме алгоритми, които да включват в логиката си приоритет, като например: винаги обхождай първо най-ляв (или най-десен) клон.

Нормална форма на чомски

КСГ $G = \langle \Sigma, V, S, R \rangle$ е в нормална форма на Чомски (НФЧ), ако всички правила са от вида $A \to BC$ или $A \to a$.

Така дефинираната граматика няма да е способна да възпроизведе празната дума ε. Следователно контекстно-свободните езици, които съдържат тези думи не могат да се генерират от КСГ в НФЧ. Но това е единствената загуба, която идва с преминаването на дадена КСГ в НФЧ.

Теорема. За всяка КСГ G, съществува КСГ G_C в НФЧ, за която $\mathscr{L}(G_C) = \mathscr{L}(G) \backslash \{\varepsilon\}$. Освен това, конструирането на G_C може да се направи с полиномиална сложност по време по отношение на размера на G. Под размер на КСГ G разбираме дължината на думата образувана от конкатенацията на всички десни части на правилата R на G.

Доказателство. Ще покажем как да трансформираме произволна КСГ $G = \langle \Sigma, V, S, R \rangle$ в НФЧ. Има три начина, по които дясната страна на дадено правило от R може да наруши органичениятa от НФЧ: *дълги правила* (тези, които имат дясна страна с повече от 2 символа), правила водещи до празната дума (*ε-правила*) и преименуващи *правила* (тези, които са от вида $A \to B$). Ще покажем как може да премахнем всяко едно от тези нарушения едно по едно.

1) **Дълги правила**. Нека $A \to B_1B_2 \ldots B_n \in R$, където $B_1, B_2, \ldots, B_n \in V \cup \Sigma$ и $n \geq 3$. Заменяме това правило с $n-1$ нови правила:

$$\begin{gathered} A \to B_1A_1, \\ A_1 \to B_2A_2, \\ \ldots \\ A_{n-2} \to B_{n-1}B_n, \end{gathered}$$

където $A_1, \ldots, A_{n-2}$ са новите нетерминали, които не са били преди това във V. Тъй като правилото $A \to B_1B_2 \ldots B_n$ може да бъде имитирано от нововъдените правила и това е единствния начин, по който нововъведените правила могат да бъдат използвани, то е ясно, че новополучената КСГ е еквивалентна на първоначалната. Прилагаме този метод на разбиване за всяко дълго правило в граматиката. Получената граматика е еквивалентна на първоначалната и има правила с дължина на дясната част не по-голяма от 2.

Пример: Да разгледаме граматиката, която генерира всички балансирани изрази от скоби, която разгледахме по-рано с правила $S \to SS$, $S \to (S)$, $S \to \varepsilon$. Има само едно дълго правило $S \to (S)$. Заменяме го с две правила $S \to (S_1$ и $S_1 \to S)$.

2) **ε-правила**. Първо трябва да определим множеството нетерминали, които довеждат до празната дума – $\mathscr{E} = \{A \in V : A \Rightarrow^* \varepsilon\}$. Това може да направим, чрез следната процедура:

1. $\mathscr{E} \leftarrow \{\varepsilon\}$
2. докато има правило $A \to \gamma$ с $\gamma \in \mathscr{E}^*$ и $A \notin \mathscr{E}$
3. $\mathscr{E} \leftarrow \mathscr{E} \cup A$

След като вече имаме множеството $\mathscr{E}$, изтриваме от G всички ε-правила и повтаряме следната процедура: За всяко правило от вида $A \to BC$ или $A \to CB$ с $B \in \mathscr{E}$ и $C \in V \cup \Sigma$, добавяме правилото $A \to C$ към граматиката. Всяка дума, която може да се генерира от първоначалната граматика, може да се симулира и от новосъздадената и обратно, като правим само едно изключение: ε не може да се генерира от езика, тъй като вече сме изтрили правиото $A \to \varepsilon$. Но твърдението на теоремата позволява това изключение.

Пример (продължаваме примера с предходната граматиката): Имаме граматика с правила

$$S \to SS, \quad S \to (S_1, \quad S_1 \to S), \quad S \to \varepsilon$$

Прилагаме процедурата описана по-горе. Първоначално инициализираме $\mathscr{E} \leftarrow \{\varepsilon\}$. След това $\mathscr{E} = \{S, \varepsilon\}$, тъй като имаме правилото $S \to \varepsilon$. Това е и финалното множество

$\mathscr{E}$. Премахваме ε-правилата от граматиката (в нашия случай това е само $S \to \varepsilon$) и добавяме всички техни варианти, в които сме премахнали един или няколко терминала от $\mathscr{E}$(в случая ще е само един, тъй като първо сме премахнали дългите правила, за да избегнем експоненциалността от тази стъпка). Новите правила са

$$S \to SS, \quad S \to (S_1, \quad S_1 \to S), \quad S \to S, \quad S_1 \to)$$

Правилото $S \to S$ е добавено, тъй като имаме правило $S \to SS$ със $S \in \mathscr{E}$. То е несъществено и може да бъде премахнато. Правилото $S_1 \to)$ е добавено, тъй като имаме правило $S_1 \to S)$ със $S \in \mathscr{E}$.

3) **Преименуващи правила**. Граматиката ни сега има само правила с дължина на дясната страна равна на 1 или 2. Трябва да премахнем преименуващите правила, които са с дължина 1 и са от вида $A \to B$. Ще постигнем това по следния начин: За всеки символ $A \in V$ изчисляваме, чрез процедурата по-долу, множеството от символи $\mathscr{D}(A) = \{B \in V : A \Rightarrow^* B\}$, които може да се генерират от A в граматиката.

1. $\mathscr{D}(A) \leftarrow \{A\}$
2. докато има правило $B \to C$ с $B \in \mathscr{D}(A)$ и $C \notin \mathscr{D}(A)$
3. $\mathscr{D}(A) \leftarrow \mathscr{D}(A) \cup C$

Забележете, че за всеки символ A, $A \in \mathscr{D}(A)$, а ако a е терминален, тогава $\mathscr{D}(a) = \{a\}$.

Третата последна стъпка от трансформацията на нашата КСГ до такава в НФЧ е да премахнем всички преименуващи правила и да заменим всички правила от вида $A \to BC$ с всички възможни комбинации от правила във вида $A \to B'C'$, където $B' \in \mathscr{D}(B)$ и $C' \in \mathscr{D}(C)$. Такова правило имитира оригиналното правило $A \to BC$, с редица от преименуващи правила, които генерират B' от B и C' от C. Накрая добавяме правило $S \to BC$ за всяко правило $A \to BC$, за което $A \in \mathscr{D}(S) \backslash \{S\}$.

Отново получената граматика е еквивалентна на първоначалната, преди премахването на преименуващите правила, тъй като ефекта на преименуващите правила се имитира, което се гарантира от добавянето на $S \to BC$ в последната стъпка.

Пример (продължение): В нашата модифицирана граматика имаме правилата:

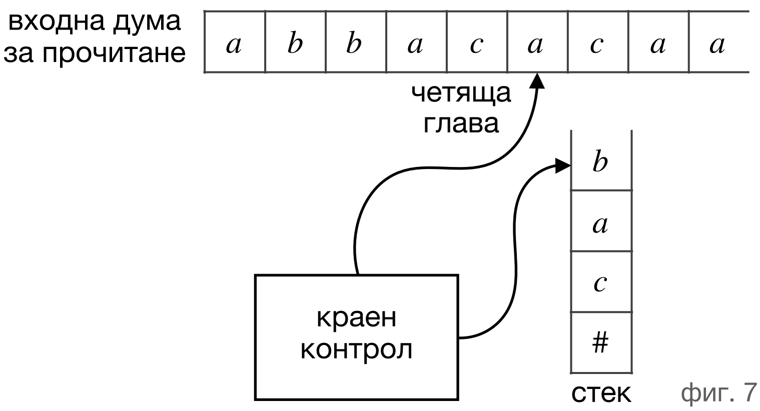
$$S \to SS, \quad S \to (S_1, \quad S_1 \to S), \quad S_1 \to)$$

Имаме $\mathscr{D}(S_1) = \{S_1,)\}$, и $\mathscr{D}(A) = \{A\}$ за всяко $A \in V \backslash \{S_1\}$. Премахваме всички къси правила, които в случая са само $S_1 \to)$. Единствения нетерминал с нетривиално множество $\mathscr{D}$ е S_1, които се появява в дясната страна единствено на второто правило. Следователно заменяме това правило с нови две правила $S \to (S_1$ и $S \to ()$, съответстващи на двата елемента от $\mathscr{D}(S_1)$. Окончателно, КСГ в НФЧ, еквивалентна на първоначалната КСГ е

$$S \to SS, \quad S \to (S_1, \quad S_1 \to S), \quad S \to ()$$

Остана само да докажем, че описаните конструкции няма да надвишат полиномиалната времева сложност по отношение на размера на оригиналната граматика G. Припомняме, че под размер на граматика имаме предвид сумата от дължините на десните части от правилата на G. Нека n е този размер. Премахването на дългите правила отнема $O(n)$ време и образува граматика с размер $O(n)$. Премахването на ε-правилата отнема $O(n^2)$ време за изпълнението на процедурата ($O(n)$ итерации, $O(n)$ време за всяка) плюс $O(n)$ за добавянето на нови правила. Накрая, премахването на късите правила отново не надвишава полиномиалното време ($O(n)$ итерации, по $O(n^2)$ време за всяка).

Стекови автомати



фиг. 7

Преход в стековия автомат:

$$\Delta(\overset{\text{текущо състояние}}{q},\ \overset{\text{буква за прочитане}}{a},\ \overset{\text{текущ връх на стека}}{b}) = (\overset{\text{ново състояние}}{p},\ \overset{\text{нов връх на стека}}{c})$$

Някой видове преходи (операции), с които да придобием представа как работи стековият автомат:

$$\Delta(q, a, b) = \begin{cases} (p, \varepsilon) & \text{— изтрива върха на стека — } pop() \\ (p, Ab) & \text{— добавя } A \text{ в стека — } push(A) \\ (p, A) & \text{— замества върха на стека с } A \end{cases}$$

$$\Delta(q, a, \varepsilon) = \begin{cases} (p, b) & \text{— слага } b \text{ в празен стек} \\ \dots \\ \dots \end{cases}$$

$$\Delta(q, \varepsilon, A) = \begin{cases} (p, A) & \text{— ако върха на стека е } A \text{ преминава в състояние } p \text{ без да прочита нищо} \\ \dots \\ \dots \end{cases}$$

Недетерминиран стеков автомат (НСА) Push-down automaton (PDA)

НСА наричаме седморката $P = \langle Q, \Sigma, \Gamma, \#, q_{\text{start}}, q_{\text{accept}}, \Delta \rangle$, където:

1) Q – крайно множество от състояния;
2) Σ – крайна входна азбука;
3) Γ – крайна стекова азбука (тук няма изискване както при граматиките $\Sigma \cap \Gamma = \emptyset$);
4) # – специален символ за дъно на стека, $\# \notin \Sigma \cup \Gamma$;
5) q_{start} – началното състояние;
6) q_{accept} – финалното състояние (единственото финално състояние не променя изразителната сила на стековия автомат);
7) Δ – релацията на преходите, която е подмножество на $(\overset{\text{текущо състояние}}{Q} \times \overset{\text{буква за прочитане}}{(\Sigma \cup \{\varepsilon\})} \times \overset{\text{текущ връх на стека}}{\Gamma}) \times (\overset{\text{ново състояние}}{Q} \times \overset{\text{нов връх на стека}}{\Gamma^{\leq 2}})$.

Някои важни неща, които трябва да отбележим за сигнатурата на Δ: буквата за прочитане може да е буква от азбуката или празната буква, но задължително трябва да имаме буква в стека, която да прочетем, за да може да извършим преход.
$\Gamma^{\leq 2} = \Gamma^0 \cup \Gamma^1 \cup \Gamma^2$; $\Gamma^0 = \{\varepsilon\}$ – изтрива върха на стека, $\Gamma^1 = \{a\}$ – заменя върха на стека; $\Gamma^2 = \{ab\}$ – добавя в стека.

Конфигурация на стеков автомат дефинираме като елемент на $Q \times \Sigma^* \times \Gamma^*$, където първата компонента е състоянието, в което се намира текущото изчисление на входната дума, втората компонента е остатъка от входната дума за прочитане, а третата компонента описва какво представлява целия стек. Конфигурацията представлява моментно изчерпателно описание на стековия автомат.

Дефиниция ($\vdash_P$). Релацията $\vdash_P$ върху автоматна конфигурация:

$$\frac{\Delta(q, a, A) \ni (p, \beta)}{(q, a\delta, A\gamma) \vdash_P (p, \delta, \beta\gamma)}$$

Тоест, ако съществува преход с текущо състояние q, който прочита буквата a с текущ връх на стека A и отива в ново състояние p като актуализира върха на стека с β, тогава автомата преминава от конфигурация $(q, a\delta, A\gamma)$ за една стъпка в конфигурация $(p, \delta, \beta\gamma)$

или

$$\frac{\Delta(q, \varepsilon, A) \ni (p, \beta)}{(q, \delta, A\gamma) \vdash_P (p, \delta, \beta\gamma)}$$

Тоест, в този случай не прочитаме нищо от входната дума, но въпреки това правим операция в автомата и преминаваме от едно в друго състояние като променяме върха на стека.

Така релацията вече е напълно дефинирана и ни казва как от една конфигурация на изчисление на стековия автомат преминава в друга конфигурация чрез една стъпка.

Релацията $\vdash_P^*$ е рефлексивното (вкл. 0 стъпки) и транзитивно затваряне на релацията $\vdash_P$. Тази релация ни казва до каква конфигурация е доведен автомата след краен брой стъпки от дадена начална конфигурация.

Дефиниция (език на стеков автомат). Език на стеков автомат дефинираме по следния начин: $\mathscr{L}(P) = \{\omega \in \Sigma^* \,|\, (q_{\text{start}}, \omega, \#) \vdash_P^* (q_{\text{accept}}, \varepsilon, \varepsilon)\}$.

Да разгледаме класическият пример за език, който не е регулярен но е контекстно-свободен.

$$L = \{a^n b^n \,|\, n \in \mathbb{N}_0\}, S \to aSb \,|\, \varepsilon$$

Ще дефинираме НСА P, такъв че $\mathscr{L}(P) = L$.
$Q = \{q, p, f\}, q_{\text{start}} = q, q_{\text{accept}} = f$.

1. $\Delta(q, \varepsilon, \#) = \{(f, \varepsilon)\}$
2. $\Delta(q, a, \#) = \{(q, a\#)\}$
3. $\Delta(q, a, a) = \{(q, aa)\}$
4. $\Delta(q, b, a) = \{(p, \varepsilon)\}$
5. $\Delta(p, b, a) = \{(p, \varepsilon)\}$
6. $\Delta(p, \varepsilon, \#) = \{(f, \varepsilon)\}$

Оказа се, че автоматът е детерминиран, тъй като от дясната страна имаме само singleton-и. Всички останали троики, които не участват в изброените допустими операции, са от вида $\Delta(r, x, A) = \emptyset$. (Детерминиран е, тъй като по естествен начин думите от езика са от тякъв вид, че подсказват на автомата кога да превключи състоянията.)

Нека $\omega_1 = a^2b^2 \in \mathscr{L}(P)$:

$$(q, a^2b^2, \#) \overset{2.}{\vdash} (q, ab^2, a\#) \overset{3.}{\vdash} (q, b^2, aa\#) \overset{4.}{\vdash} (p, b, a\#)$$
$$\overset{5.}{\vdash} (p, \varepsilon, \#) \overset{6.}{\vdash} (f, \varepsilon, \varepsilon) \overset{\text{def.}}{\Rightarrow} \omega_1 \in \mathscr{L}(P)$$

Нека $\omega_2 = a^2b^3 \notin \mathscr{L}(P)$:

$$(q, a^2b^3, \#) \vdash^* (q, b^3, a^2\#) \overset{4.}{\vdash} (p, b^2, a\#) \overset{5.}{\vdash} (p, b, \#)$$
$$\overset{6.}{\vdash} (f, b, \varepsilon) \overset{\text{def.}}{\Rightarrow} \omega_2 \notin \mathscr{L}(P)$$

Нека $\omega_3 = \varepsilon \in \mathscr{L}(P)$:

$$(q, \varepsilon, \#) \overset{1.}{\vdash} (p, \varepsilon, \varepsilon) \Rightarrow \omega_3 \in \mathscr{L}(P)$$

Пример за език, при който стековият автомат е недетерминиран.

$$L = \{\omega\omega^{rev} \,|\, \omega \in \{a, b\}^*\}, S = aSa\,|\,bSb\,|\,\varepsilon$$

$Q = \{q, p, f\}, q_{\text{start}} = q, q_{\text{accept}} = f.$

$$\begin{cases} 1.\ \Delta(q, \varepsilon, \#) = \{(f, \varepsilon)\} \\ \begin{cases} 2.\ \Delta(q, a, \#) = \{(q, a\#)\} \\ 3.\ \Delta(q, b, \#) = \{(q, b\#)\} \end{cases} \\ \begin{cases} 4.\ \Delta(q, a, b) = \{(q, ab)\} \\ 5.\ \Delta(q, b, a) = \{(q, ba)\} \end{cases} \\ \begin{cases} 6.\ \Delta(q, a, a) = \{(q, aa), (p, \varepsilon)\} \\ 7.\ \Delta(q, b, b) = \{(q, bb), (p, \varepsilon)\} \end{cases} \\ \begin{cases} 8.\ \Delta(p, a, a) = \{(p, \varepsilon)\} \\ 9.\ \Delta(p, b, b) = \{(p, \varepsilon)\} \end{cases} \end{cases}$$

Нека $\omega_1 = abbbba$

$$(q, abbbba, \#) \vdash^* (q, bba, bba\#) \overset{7.}{\vdash} (p, ba, ba\#)$$
$$\vdash^* (p, \varepsilon, \#) \overset{1.}{\vdash} (f, \varepsilon, \varepsilon) \overset{\text{def.}}{\Rightarrow} \omega_1 \in \mathscr{L}(P)$$

Нека $\omega_2 = abab$

$$(q, abab, \#) \overset{2.}{\vdash} (q, bab, a\#) \overset{5.}{\vdash} (q, ab, ba\#) \overset{4.}{\vdash} (q, b, aba\#)$$
$$\overset{5.}{\vdash} (q, \varepsilon, baba\#) \overset{\text{def.}}{\Rightarrow} \omega_2 \notin \mathscr{L}(P)$$

Коментар: За езика $\tilde{L} = \{\omega\$\omega^{rev} \,|\, \omega \in \{a, b\}^*\}$ има ДСА, тъй като сепаратора $\$$ му подсказва кога да превключи състоянията при изчислението.

Всеки краен автомат може да бъде тривиално разглеждан като стеков автомат, който никога не оперира със стека си. Нека $A = \langle Q, \Sigma, \Delta, s, F\rangle$ е недетерминиран краен автомат и нека $P = \langle Q, \Sigma, \Gamma, \#, q_{\text{start}}, q_{\text{accept}}, \tilde{\Delta}\rangle$, където

$$\tilde{\Delta} = \{((p, a, \varepsilon), (q, \varepsilon)) : (p, a, q) \in \Delta\}.$$

Казано с други думи, P имитира преходите на A. Очевидно A и P приемат един и същ език.

Връзка между стековите автомати и контекстно-свободните граматики (Възможно е да се даде само тази част от въпроса!)

Теорема. Множеството от езиците генерирани от КСГ и множеството от езиците разпознавани от стекови автомати съвпадат.

Ще докажем теоремата в едната посока, а именно, че **за всеки език генериран от КСГ съществува стеков автомат, който го разпознава.**

Доказателство. Нека $G = \langle \Sigma, V, S, R \rangle$ е КСГ. Трябва да построим стеков автомат P, такъв, че $\mathscr{L}(P) = \mathscr{L}(G)$. Б.о.о. приемаме, че G е в НФЧ, тъй като всяка КСГ може да се приведе в такъв вид. Строим $P = \langle Q, \Sigma, \Gamma, \#, q_{\text{start}}, q_{\text{accept}}, \Delta \rangle$, където

$Q = \{q_{\text{start}}, p, q_{\text{accept}}\}$

начално състояние — работно състояние — финално състояние

$\Gamma = \Sigma \cup V \cup \{\#\}, \# \notin \Sigma \cup V.$

$$\Delta = \begin{cases} 1.\ \Delta(q_{\text{start}}, \varepsilon, \#) = \{(p, S\#)\} \\ 2.\ \Delta(p, \varepsilon, A) = \{(p, \beta) \mid A \to \beta \text{ е правило в } G\} \\ 3.\ \Delta(p, a, a) = \{(p, \varepsilon)\} \\ 4.\ \Delta(p, \varepsilon, \#) = \{(q_{\text{accept}}, \varepsilon)\} \end{cases}$$

1. не прочитаме нищо от входа при празен стек, добавяме началния нетерминал на граматиката и преминаваме в работно състояние.

2. всичко е коректно, тъй като граматиката е в НФЧ $\Rightarrow |\beta| \leq 2$ и дефиницията на Δ е удовлетворена

Пример (примера трябва да е в НФЧ):

$S \to AS \mid AB$

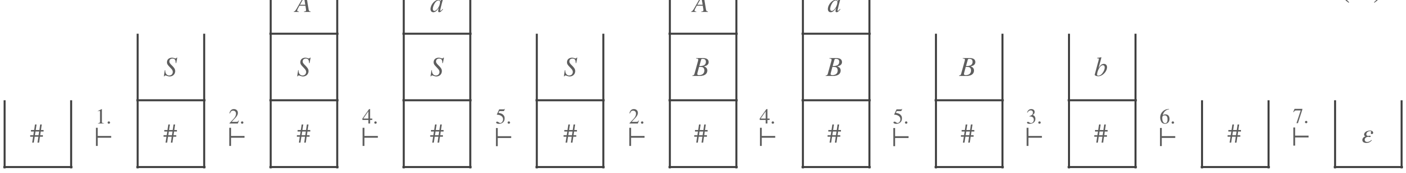
$B \to AB \mid b$

$A \to a$

$$\begin{cases} 1.\ \Delta(q_{\text{start}}, \varepsilon, \#) = \{(p, S\#)\} \\ 2.\ \Delta(p, \varepsilon, S) = \{(p, AS), (p, AB)\} \\ 3.\ \Delta(p, \varepsilon, B) = \{(p, AB), (p, b)\} \\ 4.\ \Delta(p, \varepsilon, A) = \{(p, a)\} \\ 5.\ \Delta(p, a, a) = \{(p, \varepsilon\} \\ 6.\ \Delta(p, b, b) = \{(p, \varepsilon)\} \\ 7.\ \Delta(p, \varepsilon, \#) = \{(q_{\text{accept}}, \varepsilon)\} \end{cases}$$

$S \Rightarrow AS \Rightarrow aS \Rightarrow aAB \Rightarrow aaB \Rightarrow aab$, т.е. $S \Rightarrow^* aab$

$\Rightarrow aab \in \mathscr{L}(P)$



Трябва да докажем, че за така построения НСА $\mathscr{L}(G) = \mathscr{L}(P)$ в общия случай. За целта трябва да докажем двете страни на равенството. Ще докажем, че

$$\frac{S \overset{l}{\underset{left}{\Rightarrow}} \beta\gamma, \beta \in \Sigma^*, \gamma \in (V\Sigma^*)^*}{(p, \beta, S\#) \vdash^* (p, \varepsilon, \gamma)}(1) \quad \text{и} \quad \frac{(p, \beta, \delta\#) \overset{l}{\vdash} (p, \varepsilon, \#), \delta \in (V \cup \Sigma)^*}{\delta \underset{left}{\Rightarrow^*} \beta}(2)$$

Ако докажем тези две твърдения, то като частен случай ще може да вземем $\gamma = \varepsilon$ за (1) и $\delta = S$ за (2). Тогава двете твърдения ще са еквивалентни на

$$\frac{\text{Ако } \beta \in \mathscr{L}(G),}{\text{то } \beta \in \mathscr{L}(P)}(1) \quad \text{и} \quad \frac{\text{Ако }, \beta \in \mathscr{L}(P)}{\text{то } \beta \in \mathscr{L}(G)}(2)$$

Доказателство на (1).

Ще направим индукция по дължината l на извода. Ще докажем, че ако $S \overset{l}{\underset{left}{\Rightarrow}} \beta\gamma$, то ще може да прочетем думата β от автомата и да получим в стека $\gamma\#$, т.е.

$$(p, \beta, S\#) \vdash^* (p, \varepsilon, \gamma\#)$$

1.1. Индукционна база. За $l = 0$ имаме, че $S \overset{0}{\Rightarrow} S$ от където следва, че $\beta = \varepsilon$ и $\gamma = S$. Тогава верността на $(p, \varepsilon, S\#) \vdash^* (p, \varepsilon, S\#)$ е тривиална.
1.2. Индукционна хипотеза. Допускаме, че твърдението е в сила за дължина на най-ляв извод не по-голяма от $l-1, l > 0$.
1.3. Индукционен преход. Ще докажем, че е в сила и за извод с дължина l. Нека $S \overset{l}{\underset{left}{\Rightarrow}} \beta\gamma$.

Искаме да разбием този извод $\beta\gamma$ на части, за да използваме индукционното предположение. В общия случай имаме

$$\frac{S \overset{l-1}{\underset{left}{\Rightarrow}} \beta_1 A \gamma_2\,, \quad A \to \beta_2\gamma_1}{S \overset{l}{\underset{left}{\Rightarrow}} \beta_1\beta_2\gamma_1\gamma_2 \text{ така, че } \beta_1\beta_2 = \beta \text{ и } \gamma_1\gamma_2 = \gamma \text{ и } \beta_1, \beta_2 \in \Sigma^* \text{ (най-ляв извод)}}$$

От индукционната хипотеза имаме (за $\beta = \beta_1$ и $\gamma = A\gamma_2$), че $(p, \beta_1, S\#) \vdash^* (p, \varepsilon, A\gamma_2\#)$ $\vdash (p, \varepsilon, \beta_2\gamma_1\gamma_2\#)$.

$$\frac{(p, \beta_1, S\#) \vdash^* (p, \varepsilon, \beta_2\gamma_1\gamma_2\#)}{(p, \beta_1\beta_2, S\#) \vdash^* (p, \beta_2, \beta_2\gamma_1\gamma_2\#)}$$

От най-ляв извод знаем, че $\beta_2 \in \Sigma^*$, а от НФЧ следва, че $|\beta_2| \leq 2$. Следователно, $(p, \beta_1\beta_2, S\#) \vdash^* \underbrace{(p, \beta_2, \beta_2\gamma_1\gamma_2\#) \vdash^* (p, \varepsilon, \gamma\#)}$, което искахме да докажем.

Доказателство на (2).

Ще направим индукция по дължината l на изчислението на автомата. Ще докажем, че ако $(p, \beta, \delta\#) \overset{l}{\vdash} (p, \varepsilon, \#)$, то може да стигнем до най-ляв извод β в граматиката G, т.е. $\delta \underset{left}{\Rightarrow^*} \beta$.

1.1. Индукционна база. За $l = 0$ имаме, $(p, \beta, \delta\#) \overset{0}{\vdash} (p, \varepsilon, \#) \Rightarrow \beta = \delta = \varepsilon$.
1.2. Индукционна хипотеза. Допускаме, че твърдението е в сила за автоматни изчисления с дължина не по-големи от $l-1, l > 0$.

1.3. Индукционен преход. Ще докажем, че е в сила и за изчисление с дължина l.

Ще разгледаме 3 случая по отношение на първото изчисление:

I сл. $\Delta(p, a, a) = \{(p, \varepsilon)\}$
II сл. $\Delta(p, \varepsilon, A) = \{(p, BC)\}$
III сл. $\Delta(p, \varepsilon, A) = \{(p, a)\}$

I сл. Нека $\delta = a\delta'$, тогава $\beta = a\beta'$ и $(p, a\beta', a\delta'\#) \vdash \underbrace{(p, \beta', \delta'\#) \overset{l-1}{\vdash} (p, \varepsilon, \#)}_{\text{И.Х.}}$

$$\frac{\delta' \underset{left}{\Rightarrow^*} \beta' \,, \quad a \underset{left}{\overset{0}{\Rightarrow}} a}{\underbrace{a\delta'}_{\delta} \underset{left}{\Rightarrow^*} \underbrace{a\beta'}_{\beta}}$$

II сл. Нека $\delta = A\delta'$, тогава $(p, \beta, A\delta'\#) \vdash \underbrace{(p, \beta, BC\delta') \overset{l-1}{\vdash} (p, \varepsilon, \#)}_{\text{И.Х.}}$

$$\frac{\dfrac{A \to BC}{A\delta' \underset{left}{\overset{1}{\Rightarrow}} BC\delta'} \,, \quad BC\delta' \underset{left}{\Rightarrow^*} \beta}{A\delta' \underset{left}{\Rightarrow^*} \beta}$$

III сл. Нека $\delta = A\delta'$, тогава $(p, \beta, A\delta'\#) \vdash \underbrace{(p, \beta, a\delta'\#) \overset{l-1}{\vdash} (p, \varepsilon, \#)}_{\text{И.Х.}}$

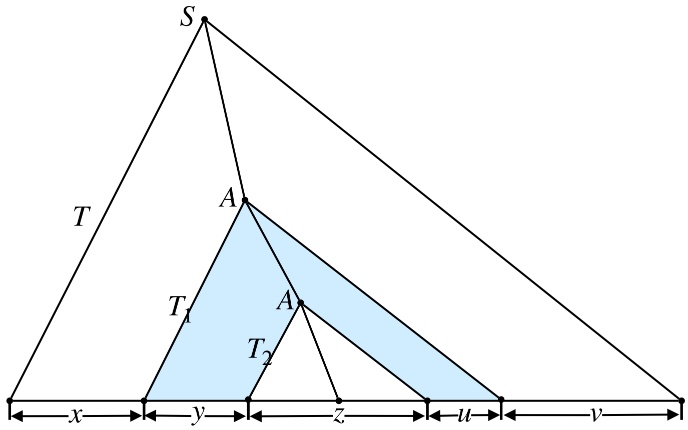
$$\frac{a\delta'\# \underset{left}{\Rightarrow^*} \beta \,, \quad \dfrac{A \to a}{A\delta' \to a\delta'}}{\underbrace{A\delta'}_{\delta} \underset{left}{\Rightarrow^*} \beta}$$

Свойства на затвореност
Контекстно-свободните езици са затворени относно обединение, конкатенация и итерация (звезда на Клини). **НЕ** са затворени относно сечение и допълнение, но са затворени относно сечение с регулярни езици.

Лема за покачването (xyzuv)
Нека $G = \langle \Sigma, V, S, R \rangle$ е КСГ с безкраен език $\mathscr{L}(G) = L$. Съществува естествено число n, такова че за всяка дума $\omega \in L$ с дължина $|\omega| \geq n$, съществуват думи x, y, z, u, v, за които $\omega = xyzuv$, $yu \neq \varepsilon$, $|yzu| \leq n$ и за всяко естествено число i, думата xy^izu^iv е от L.

Доказателство. Нека $G = \langle \Sigma, V, S, R \rangle$ е КСГ, която генерира L. Дефинираме $\Phi(G) = \max\left(\{\, |\alpha| \,\middle|\, A \to \alpha \in R\right)$ – максимална разклоненост на всички дървета на извод за G. Лесно се доказва с индукция, че за всяко дърво на извод T за G, $|\omega(T)| \leq \Phi(G)^{h(T)}$.
Нека $n = \Phi(G)^{|V|+1}$, $\omega \in L$ и $|\omega| \geq n$. Нека T е дърво на извод (за G) с най-малък брой листа, за което $\omega(T) = \omega$.
Знаем, че $\Phi(G)^{|V|+1} \leq |\omega| = |\omega(T)| \leq \Phi(G)^{h(T)}$. Следователно $h(T) \geq |V| + 1 \Rightarrow$ съществува път от корена до листо с дължина поне $|V| + 1$, т.е. в него участват поне $|V| + 2$ върха, където точно един е терминал (листото), а останалите поне $|V| + 1$ са нетерминали.

Но броят на всички нетерминали в G е $|V|$ и от принципа на Дирихле следва, че поне един нетерминал участва поне два пъти в този път в T.



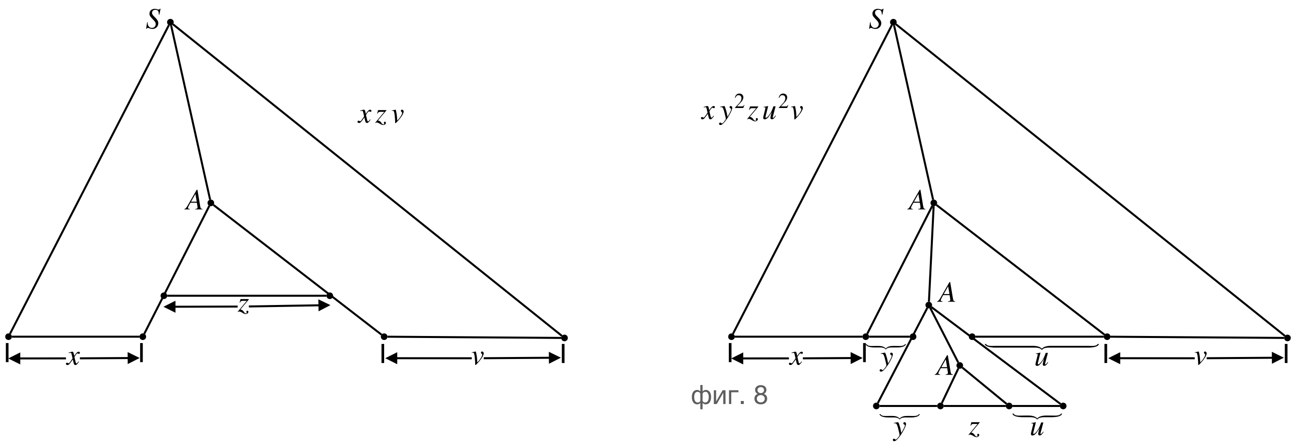
фиг. 8

Да разгледаме поддърво T_1 на T, такова че T_1 е поддърво на T с минимална височина със следните свойства:

1. $r(T_1) = A \in V$;
2. Съществува път в T_1 от $r(T_1)$ до листо, в който A се среща поне 2 пъти.

Нека $\omega(T_1) = yzu$ и $\omega(T) = \omega = xyzuv$ както е показано на фиг. 8. В сила са следните твърдения.

1) $|yu| \geq 1$, тъй като, ако $|yu| = 0$ (т.е. $yu = \varepsilon$), то може да пропуснем едно разширяване на A и да получим дърво T' с $\omega(T') = \omega$ и с по-малък брой листа от T, което е противоречие с допускането, че T има минимален брой листа измежду всички дървета G с дума ω $\Rightarrow yu \neq \varepsilon \Rightarrow |yu| \geq 1$;
2) $|yzu| \leq n$, тъй като, ако $|yzu| > n$, то $\Phi(G)^{|V|+1} = n \leq |yzu| = |\omega(T_1)| \leq \Phi(G)^{h(T_1)}$. Следователно $h(T_1) > |V| + 1 \Rightarrow h(T_1) \geq |V| + 2 \Rightarrow$ съществува път в T_1 от $r(T_1)$ до листо, в който участват поне $|V| + 2$ нетерминала, т.е. в този път или $r(T_1)$ участва поне 3 пъти или има друг нетерминал, който участва поне 2 пъти. И в двата случая има дърво T_1', което е поддърво на T_1, $h(T_1') < h(T_1)$ и T_1' има същите свойства като на T_1. Следователно T_1 не е с минимална височина от всички поддървета на T с желаните свойства – противоречие. Следователно $|yzu| \leq n$.
2) $(\forall i \in \mathbb{N})\left[xy^izu^iv \in L\right]$, тъй като може да повторим частта $A \Rightarrow \ldots \Rightarrow yAu$ произволен брой пъти (включително и 0):



фиг. 8

Примери за езици, които не са контекстно-свободни

a) $L = \{a^n b^n c^n \mid n \in \mathbb{N}\}$

b) $L = \{a^n \mid n \geq 1 \text{ е просто число}\}$; $L = \{a^{2^n} \mid n \in \mathbb{N}\}$; $L = \{a^{n^2} \mid n \in \mathbb{N}\}$

c) $L = \{(a^n b^n)^n \mid n \in \mathbb{N}\}$

d) $L = \{\omega \in \{a, b, c\}^* \mid \omega \text{ има равен брой } a, b \text{ и } c\}$

e) $L = \{\omega\omega\omega \mid \omega \in \{a, b\}^*\}$

Доказателство за пример а). Да допуснем, че $L = \mathscr{L}(G)$, за някоя КСГ $G = \langle \Sigma, V, S, R \rangle$. Нека $n > \dfrac{b^{|V|}}{3}$. Тогава $a^n b^n c^n$ е в $\mathscr{L}(G)$ и има представяне $\omega = xyzuv$, за което $yz \neq \varepsilon$ (поне една от тян не е празната дума) и $xy^izu^iv \in \mathscr{L}(G)$ за всяко $i = 0, 1, 2, \ldots$. Има два възможни случая, които и двата ще доведат до противоречие. Ако yu съдържа всички три символа a, b, c, то поне поне една от думите v или y трябва да съдържа поне два от тях. Но тогава xy^2zu^2v съдържа появяване на букви, които не са в правилния ред: b преди a или c преди a или b. Ако в yu се появяват някои но не всички букви, тогава xy^2zu^2v няма да има равен брой от буквите a, b и c.