Най-напред ще видим как се прилага лемата за разрастването за КСЕ. Да разгледаме езикът $L=\{a^n b^n c^n \mid n\in N\}$. Да допуснем, че $L \in$ КСЕ. Тогава съществува КСГ $G=(V,\Sigma,R,S)$, която поражда този език L, т.е. $L=L(G)$. Съгласно лемата за разрастването да си изберем $n > \Phi(G)^{|V\setminus\Sigma|}$. За такова n да си вземем $w=a^n b^n c^n$, т.е. $\exists$ $3n > \Phi(G)^{|V\setminus\Sigma|}$ $w\in L$ и $|w| > \Phi(G)^{|V\setminus\Sigma|}$. Съгласно лемата w може да се представи във вида $w=uvxyz$, така че $vy\neq\varepsilon$ и $uv^kxy^kz\in L=L(G)$ за всяко естествено k.

I случай vy съдържа само една буква от a,b,c. Нека например съдържа само буквата a. Тогава $y=a^q$, $v=a^p$, $p,q\geq 0$ и $p+q>0$ защото $vy\neq\varepsilon$. В такъв случай $u=a^{r_1}$, $x=a^{r_2}$, $z=a^{r_3}b^nc^n$. Според лемата думата $w_k=uv^kxy^kz\in L$, $w_k=a^{r_1+r_2+r_3+(q+p)k}.b^n.c^n$, $r_1+r_2+r_3+k(p+q)=n$. Тъй като $p+q>0$, то за достатъчно големи k $r_1+r_2+r_3+k(p+q)>n$ и, следователно, $w_k\notin L$ за достатъчно големи k. Аналогично ще получим противоречие и ако vy съдържат само b или само c. Следователно I случай е невъзможен.

II случай. vy съдържат две букви, например a и b. Да отбележим, че не е възможно v да съдържа и a и b или y да съдържа и a и b. Причината е, че ако $v=a^p.b^q$, $p>0$ и $q>0$, то $v^2=a^p.b^q.a^p.b^q$, т.е. думата $w_2=uv^2xy^2z$ ще съдържа b-та преди a-та, т.е. $w_2\notin L$. Аналогично и за y. Следователно ще имаме $v=a^p$ $y=b^q$. Тогава отаче с аналогични разсъждения, както в първия случай за $p>0$ ще видим, че за достатъчно големи k, kp ще расте към $+\infty$ и $kp>n$ и следователно $w_k\notin L$. Ако $q>0$, аналогично като $w_k=a^{r_1}.a^{kp}.a^{r_2}b^{s_1}b^{qk}.b^{s_2}.c^n\notin L$, т.е. или a-тата или b-тата ще са повече на брой от c-тата. Следователно II случай е невъзможен.

III случай. vy съдържа трите букви a,b,c. Този слу-

чай може да се сведе до случай, в който или има в-та преди а-та или да има с-та преди в-та. Следователно и III случай е невъзможен.
Така допускането, че L е КСЕ ни доведе до противоречие. Следователно L не е КСЕ.
Лемата за разрастването може да се формулира и по следния начин:
Нека $L \subseteq \Sigma^*$, L е КСЕ. Тогава съществува ест. число n, такова, че за всяка дума $w \in L$, $|w| \geq n$ съществуват думи u, v, x, y, z, такива, че $w = uvxyz$ и $|vxy| \leq n$, $vy \neq \varepsilon$ и за всички $i \in \mathbb{N}$ е изпълнено $uv^ixy^iz \in L$.
Тази формулировка следва директно от доказаната формулировка, като положим $n = \Phi(G)^{|V - \Sigma|} + 1$ и в доказателството на лемата си изберем в дървото T'' с корен A да няма повтарящи се върхове. Това гарантира новото условие $|vxy| \leq n$.
Преди да докажем следващото твърдение да разгледаме следните два примера.
Пример 1. Езикът $L_1 = \{a^n b^n c^m \mid m, n \in \mathbb{N}\}$ е КСЕ.
Пример 2. Езикът $L_2 = \{a^m b^n c^n \mid m, n \in \mathbb{N}\}$ е КСЕ.
За пример 1 да разгледаме граматиката $G_1 = (V_1, \Sigma, R_1, S_1)$, където $V_1 = \Sigma \cup \{S_1, C\}$, $R_1 = \{S_1 \to \varepsilon, S_1 \to S_1 C, S_1 \to a S_1 b, C \to \varepsilon, C \to Cc\}$, а за пример 2 – граматиката $G_2 = (V_2, \Sigma, R_2, S_2)$, където $V_2 = \Sigma \cup \{A, S_2\}$, $R_2 = \{S_2 \to \varepsilon, S_2 \to A S_2, S_2 \to b S_2 c, A \to aA, A \to \varepsilon\}$. Лесно се проверява, че $L(G_1) = L_1$ и $L(G_2) = L_2$.
Твърдение. Множеството на КСЕ не е затворено относно операциите сечение и допълнение.
Доказателство. Да забележим най-напред, че $L = L_1 \cap L_2$, където $L = \{a^n b^n c^n \mid n \in \mathbb{N}\}$ не е КСЕ, както доказахме по-горе. От друга страна L_1 и L_2 са контекстно своб. С това посочихме пример на КСЕ, чието сечение не е КСЕ, с което е доказана първата част на твърдението. Да напомня, че за произволен език L с $\overline{L}$ озна-

чаваме допълнението му, т.е. $\overline{L'} = \Sigma^* \setminus L'$. Тогава използувайки законите на ДеМорган за множества имаме $\overline{L_1 \cap L_2} = \overline{L_1} \cup \overline{L_2}$, т.е. $L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_2}}$. Ако допуснем, че допълнението запазва КСЕ ще излезе, че $\overline{L_1}$ и $\overline{L_2}$ са КСЕ, а от твърдението от предната лекция имаме, че $\overline{L_1} \cup \overline{L_2}$ е КСЕ, а оттам и $\overline{\overline{L_1} \cup \overline{L_2}}$ е КСЕ, т.е $L_1 \cap L_2$ е КСЕ. Това е противоречие, което се дължи на допускането, че допълнението запазва КСЕ. Следователно КСЕ не са затворени отн. опер. допълнение.

Нормални форми на Чомски

Д. Казваме, че една КСГ $G = (V, \Sigma, R, S)$ се намира в нормална форма на Чомски ако $R \subseteq (V \setminus \Sigma) \times V^2$, т.е. всички правила са от вида $A \to BC$, където $A \in V \setminus \Sigma$, а $B, C \in V$.

Искам да обърна внимание, че понякога, в някои учебници КСГ се нарича, че е в нормална форма на Чомски, ако освен горните правила има и правила $A \to a$ и $A \to \varepsilon$, за $A \in V \setminus \Sigma$ и $a \in \Sigma$. С цел приложенията, които имаме предвид, ние няма да използваме този вариант на дефиницията.

Теорема. За всяка КСГ G съществува КСГ G' в нормална форма на Чомски, такава че $L(G') = L(G) \setminus (\Sigma \cup \{\varepsilon\})$.

С други думи, това което се поражда в G' и това, което се поражда в G евентуално се различава с краен брой думи, които са еднобуквени или ε.

Д-во. Нашата цел е да намерим граматика G' чиито правила са от вида $A \to BC$, където $A, B, C \in V$. За тази цел, всички правила от G, които не са от този вид ще ги модифицираме в еквивалентни правила от вида $A \to BC$. Тези правила, които ще модифицираме са от 3 вида: така наречените дълги правила от вида $A \to x$, където $|x| > 2$, така наречените

къси правила, които имат вида $A \to a$ и $A \to B$, където $A, B \in V \setminus \Sigma$ и $a \in \Sigma$ и така наречените ε-правила от вида $A \to \varepsilon$, $A \in V \setminus \Sigma$.
Подобно на случая, когато за стеков автомат построихме прост стеков автомат на стъпки, ние ще построим КСГ G' в нормална форма на Чомски на стъпки: I стъпка – ще елиминираме дългите правила като ги заменим с „нормални"; II стъпка – ще елиминираме ε-правилата, като ги заменим с „нормални" или къси правила; III стъпка – ще заменим късите правила с „нормални".
Нека $G = (V, \Sigma, R, S)$ е произволна КСГ. Ще построим нова КСГ $G' = (V', \Sigma, R', S)$, където V' ще е разширение на V с нови нетерминали, а в R' ще заменим само дългите, ε-правилата и късите правила с „нормални" правила. Под „нормални" правила разбираме такива, които са в нормална форма на Чомски.
I стъпка. Нека $A \to B_1 \dots B_n$ е дълго правило от R, такова, че $n > 2$, $A \in V \setminus \Sigma$ и $B_1, \dots, B_n \in V$. Заменяме това правило със следния списък от нови правила:
$A \to B_1 A_1$, $A_1 \to B_2 A_2$, ..., $A_{n-2} \to B_{n-1} B_n$, където $A_1, \dots, A_{n-2}$ са нови нетерминални символи, които добавяме към V'. В тази стъпка повтаряме този процес за всяко дълго правило, докато изчерпим всички дълги правила.
II стъпка: Да означим с $\mathcal{E} = \{A \mid A \in V \setminus \Sigma$ и $A \Rightarrow^* \varepsilon\}$, т.е. множеството на всички нетерминали, които могат да бъдат „изтрити". Пораждането на това множество $\mathcal{E}$ може да бъде описано алгоритмично както следва:

$\mathcal{E} := \emptyset$
while съществува правило $A \to \alpha$, $\alpha \in \mathcal{E}^*$, но $A \notin \mathcal{E}$
do добави A към $\mathcal{E}$.

След като имаме получедното $\mathcal{E}$ ние изтриваме всички ε-правила и добавяме за всички правила от вида $A \to BC$ или $A \to CB$, такива, че $B \in \mathcal{E}$ и $C \in V$, ново правило от вида $A \to C$. Да обърнем внимание, че всяко правило $A \to C$, което сме добавили, може да бъде изведено от граматиката G. Има обаче нещо ново тук при изтриването на ε-правилата. Ние може да изгубим възможността да изведем $S \Rightarrow^* \varepsilon$. Така, че може да изгубим в $L(G')$ думата ε. В първата стъпка ние не губим нищо. Едва тук губим евентуално ε.

III стъпка. Остава да премахнем само късите правила, които още се наричат преименуваща такива. С $\mathcal{D}(A)$ означаваме множеството на всички символи B, които могат да бъдат изведени в граматиката от символа $A \in V$, т.е. $\mathcal{D}(A) = \{B \mid B \in V$ и $A \Rightarrow^* B\}$

Алгоритмично това може да се опише така:

$\mathcal{D}(A) := \{A\}$
while съществува правило $B \to C$, такова, че $B \in \mathcal{D}(A)$
и $C \notin \mathcal{D}(A)$
do добави C към $\mathcal{D}(A)$.

Да обърнем внимание, че ако $A = a$ — терминал, то $\mathcal{D}(A) = \{a\}$

Сега ние махаме всички къси правила и ги заместваме с ново множество от правила: Всяко правило от вида $A \to BC$ с всички възможни правила от вида $A \to B'C'$, където $B' \in \mathcal{D}(B)$ и $C' \in \mathcal{D}(C)$. Така ние симулираме всички възможни изводи от $A \to BC$ с преименуващите правила. Най-накрая добавяме правила $S \to BC$, за всяко правило $A \to BC$ такива, че $A \in \mathcal{D}(S) \setminus \{S\}$. С това завършихме „нормализирането" на правилата от G в правила на G'. В третата стъпка при всяко правило $A \to a$, $A \in V \setminus \Sigma$, $a \in \Sigma$, като го махнем, можем да загубим извод на буквата a

от Σ. Така доказахме теоремата.
Пример. Да разгледаме КСГ $G=(V,\Sigma,R,S)$, където $\Sigma=\{(,)\}$, $V=\Sigma\cup\{S\}$, $R=\{S\to\varepsilon, S\to SS, S\to(S)\}$
I стъпка: $S\to(S_1, S_1\to S)$ заменя $S\to(S)$ - ед. дълго правило, така правилата стават $S\to\varepsilon$, $S\to SS$, $S\to(S_1$, $S_1\to S)$.
II стъпка $\mathcal{E}:=\emptyset$. Тъй като има единствено правило $S\to\varepsilon$, то $\mathcal{E}=\{S\}$. Следователно ще премахнем $S\to\varepsilon$ и ще добавим правилата $S_1\to)$, $S\to S$ и получаваме правилата $S\to SS$, $S\to(S_1$, $S\to S$, $S_1\to)$, $S_1\to S)$.
Правилото $S\to S$ е безсмислено и можем да го махнем така остават правилата $S\to SS$, $S\to(S_1$, $S_1\to)$, $S_1\to S)$.
III стъпка $\mathcal{D}(S)=\{S\}$, $\mathcal{D}(()=\{(\}$, $\mathcal{D}())=\{)\}$, $\mathcal{D}(S_1)=\{S_1,)\}$
Единственото правило, което трябва да махнем е $S_1\to)$
Ще го заменим с правилата $S\to(S_1$, $S\to()$
Окончателно получаваме правилата на новата граматика: $S\to SS$, $S\to(S_1$, $S\to()$, $S_1\to S)$.

Алгоритми за КСГ

Сега ще видим теорема за КСГ, която дава полиномиална сложност на няколко алгоритъма
Теорема. а) Съществува полиномиален алгоритъм, който по дадена контекстно свободна граматика дава стеков автомат, еквивалентен на граматиката;
Забележка: Един стеков автомат M е еквивалентен на КСГ G т.т.к. $L(M)=L(G)$.
б) Съществува полиномиален алгоритъм, който по даден стеков автомат дава КСГ, еквивалентна на стековия автомат;
в) Съществува полиномиален алгоритъм, който по дадена КСГ G и дума x дава отговор дали $x\in L(G)$.
Д-во. Преди да започнем доказателството, да поясним каква е „размерността" на КСГ, т.е относно какво ще говорим за сложност. Тук под „размерност" на КСГ ще разбираме сумата от дължините на правилата в КСГ, т.е. ако $A\to x$ е правило, то $|A|+|x|$ е д-на правилото.

и събиране на всички дължина на правила.
Аналогично, „размерността" на един стеков автомат е сумата от дължините на всички правила, т.е. ако $((q,a,\alpha),(p,\beta))$ е правило то неговата д-на е $1+1+|\alpha|+1+|\beta|$. След това сумираме всички д-ни.

а) Да обърнем внимание, че на всяко правило $A \to x$ от КСГ G се съпоставя правило $((q,\varepsilon,A)(q,x))$ от построения стеков автомат M или $((q,a,a),(q,x))$ за всяко $a\in\Sigma$. Следователно сложността на M не надвишава $2|G|$ - размерността на G плюс $4|\Sigma|$, последното $4|\Sigma|$ се мажорира от $4|G|$, така че сложността не надвишава $8|G|^2$, т.е. тя е полиномиална.

б) Да обърна внимание най-напред, че $|K|+|\Sigma| \leq |M|$ за всеки стеков автомат. Първо да отбележим, че алгоритъма, който по даден стеков автомат $M=(K,\Sigma,\Gamma,\Delta,s,F)$ дава КСГ $G=(V,\Sigma,R,S)$ първо минаваше през построяването на прост стеков автомат, което имаше три стъпки. На първата стъпка на всяко правило от вида $((q,a,B_1\dots B_n),(p,\gamma))$ заменяхме с правила $((q,\varepsilon,B_1),(q_{B_1},\varepsilon))$, $((q_{B_1},\varepsilon,B_2),(q_{B_1B_2},\varepsilon))$, …, $((q_{B_1\dots B_{n-2}},\varepsilon,B_{n-1}),(q_{B_1\dots B_{n-1}},\varepsilon))$, $((q_{B_1\dots B_{n-1}},a,B_n),(p,\gamma))$, т.е. д-ната на началното правило най-много се утроява. Така, че на първия етап сложността е $O(|M|.3)$
На втория етап е почти същото, сложността е $O(3|M|)$, а на третия етап $O(|M|^2)$, така, че окончателно сложността на построяването на прост стеков автомат е $O(|M|^2)$, т.е. $|M'| \leq O(|M|^2)$
За построяването на новата граматика G за правилата от тип (2), за всяко правило $((q,a,B),(r,C))$ от Δ' съпоставяме правила $\langle q,B,p\rangle \to a\langle r,C,p\rangle$ за $p\in K'$, $a\in\Sigma\cup\{\varepsilon\}$, $B,C\in\Gamma\cup\{\varepsilon\}$, като на този етап имаме сложност $|M'|.|K'| \leq |M'|^2$. За построяването на правилата от тип (3) на всяко правило от типа $((q,a,B),(r,C_1C_2))$ съпоставяме правила $\langle q,B,p\rangle \to a\langle r,C_1,p'\rangle\langle p',C_2,p\rangle$ за

$p, p' \in K''$, т.е. сложността не надвишава $|M'|.|K''|^2$, т.е. сложността е $O(|M'|^3)$. На четвъртата стъпка сложността не надвишава $|\Sigma|$, т.е. сл. е $O(|M'|^2)$. Окончателно сложността е $O(|M'|^3)$, т.е. тя е полиномиална.

в) Първо да видим каква е сложността на алгоритъма за преобразуване на една граматика G в граматика G', която е в нормална форма на Чомски, така че $L(G') = L(G) \setminus (\Sigma \cup \{\varepsilon\})$. Нека $n = |G|$. Тогава за да се освободим от всички дълги правила, имахме сложност $O(n)$. За да се отървем от ε-правилата алгоритъма имаше сложност $O(n^2)$. За да се отървем от късите правила, алгоритъма на тази стъпка имаше сложност $O(n)$. Окончателно, за преобразуването на G в нормална форма на Чомски алгоритъма има сложност $O(n^2)$. Ще използваме КСГ в нормална форма на Чомски за да опишем алгоритъма (полиномиален), който да разпознава кога една дума x принадлежи на $L(G)$ за КСГ G в нормална форма на Чомски. Да посочим, че това е малко необичайно, защото пораждането на една граматика в общия случай не ни дава възможност за разпознаване, още повече за ефективно разпознаване.

Нека $G = (V, \Sigma, R, S)$ е КСГ в нормална форма на Чомски. По дадена дума $x = x_1 \dots x_n$, $x_i \in \Sigma$, $n \geq 2$ ние питаме дали $x \in L(G)$? За да отговорим на този въпрос ние анализираме всички поддуми на x дали принадлежат на $L(G)$. За тази цел определяме множеството $N[i, i+s]$, за $1 \leq i \leq i+s \leq n$, от всички символи от V от които може да бъде изведена думата $x_i \dots x_{i+s}$. Ще опишем алгоритъм, който намира това множество. Той почива на естеството на КСГ в нормална форма на Чомски и тръгва от по-късите думи (които разпознаваме дали са от $L(G)$) към разпознаване на по-дългите думи за принадлежност към $L(G)$. Този способ за описване на алгоритми и такива алгоритми са известни като динамично програмиране (динамични алгоритми).

Алгоритъм:

стъпка $s=0$
```
for i:=1 to n do N[i,i]={x_i};
for i:=1 to n do
for j:=1 to n do
    if i≠j then N[i,j]=∅;
for s:=1 to n-1 do
    for i:=1 to n-s do
        for k:=i to i+s-1 do
        if същ. правило A→BC от R такова, че B∈N[i,k]
        и C∈N[k+1,i+s] then добави A към N[i,i+s].
    Приеми x ако S∈N[1,n].
```

Ще докажем, че алгоритъмa е коректен, т.е. че наистина отговаря на изискванията, за които е построен. Подобни алгоритми сме разглеждали. Такъв беше алгоритъмът на Дейкстра за намиране на най-къси пътища от фиксиран връх на даден граф с тегла на всички ребра. Сега ще докажем следното твърдение:

Тв. За всяко естествено s, такова че $0 \leq s \leq n-1$ след s-тата итерация на горния алгоритъм за всички $i=1,\dots,n-s$ е изпълнено равенството

$$N[i,i+s]=\{A \mid A\in V \text{ и } A \Rightarrow^{*}_{G} x_i \dots x_{i+s}\}.$$

Д-во. Ще докажем твърдението с пълна индукция относно s. За $s=0$ твърдението е вярно (Да обърнем внимание, че за $x_i \in \Sigma$ еднобуквени думи, са единствените думи които могат да бъдат изведени от x_i и единствени букви, от които могат да се изведат x_i от x_i поради рефлексивност на $\Rightarrow^{*}_{G}$).

Да допуснем, че твърдението е вярно за всички s', $s'<s$ и $s>0$. Ще докажем, че твърдението е вярно и за s. Тогава за нетерминал A и думата $x_i \dots x_{i+s}$ имаме, че $A \Rightarrow^{*}_{G} x_i \dots x_{i+s}$ т.т.к. същ. $B, C \in V$ такива, че

$A \Rightarrow^* BC \Rightarrow^* x_i \dots x_{i+s}$ и съществува k, $i \leq k < i+s$ такива, че $B \Rightarrow^* x_i \dots x_k$ и $C \Rightarrow^* x_{k+1} \dots x_{i+s}$. Тук използуваме, че G е в нормална форма на Чомски.
Следователно $A \in \{A' \mid A' \in V$ и $A' \Rightarrow^* x_i \dots x_{i+s}\}$ т.т.к. същ. ест. k, $i \leq k < i+s$ и $B, C \in V$ и правило $A \to BC$ от G такива, че $B \in \{A' \mid A' \in V$ и $A' \Rightarrow^* x_i \dots x_k\}$ и $C \in \{A' \mid A' \in V$ и $A' \Rightarrow^* x_{k+1} \dots x_{i+s}\}$. Да напишем думата $x_i \dots x_k$ във вида $x_i \dots x_{i+s'}$, където $s' = k-i$ и думата $x_{k+1} \dots x_{i+s}$ във вида $x_{k+1} \dots x_{k+1+s''}$, където $s'' = i+s-k-1$. От факта, че $i \leq k < i+s$ получаваме, че $0 \leq s', s'' < s$. Следователно можем да приложим ИП и получаваме, че $N[i,k] = \{A' \mid A' \in V$ и $A' \Rightarrow^* x_i \dots x_k\}$ и $N[k+1, i+s] = \{A' \mid A' \in V$ и $A' \Rightarrow^* x_{k+1} \dots x_{i+s}\}$.
Така окончателно получаваме, че
$A \in \{A' \mid A' \in V$ и $A' \Rightarrow^* x_i \dots x_{i+s}\}$ т.т.к. същ. естествено число k, $i \leq k < i+s$ и символи $B \in N[i,k]$ и $C \in N[k+1, i+s]$ и правило $A \to BC$ от граматиката G. Тогава по описания алгоритъм ние добавяме винаги A към $N[i, i+s]$ т.е. $A \in \{A' \mid A' \in V$ и $A' \Rightarrow^* x_i \dots x_{i+s}\}$ т.т.к. $A \in N[i, i+s]$, т.е. двете множества $N[i, i+s]$ и $\{A' \mid A' \in V$ и $A' \Rightarrow^* x_i \dots x_{i+s}\}$ съвпадат, с което твърдението е доказано.
Сега $x \in L(G)$ т.т.к. $S \Rightarrow^* x$ т.т.к. $S \in N[1, n]$, с което коректността е доказана.
Накрая да обърнем внимание, че ако $|x| = n$, то сложността на алгоритъма е $O(|x|^3 |G|)$. Но един път фиксирана граматиката G на някой език G е непроменена за този алгоритъм. Следователно $|G|$ е константа и сложността на алгоритъма е $O(|x|^3)$ за всяка дума $x \in \Sigma^*$.