ДМ-2

Лекція 9

Тема 8. Теорія обчислень

Формальні мови і граматики

План лекції

- **Означення формальної мови**
- **Формальні породжувальні граматики**
- **Типи граматик (ієрархія Хомського)**
- **Дерева виведення**
- **Форми Бекуса–Наура**

Розглянемо головні поняття теорії формальних мов і теорії формальних граматик, покажемо зв'язок між граматиками й автоматами.

Формальні мови та граматики мають велике значення в побудові й реалізації мов програмування. Скінченні автомати й такі тісно пов'язані з ними конструкції, як, наприклад, регулярні граматики та регулярні вирази, належать до найважливіших понять інформатики. Різні варіанти скінченних автоматів використовують для опису й аналізу технічних пристроїв, різних систем і процесів, програм і алгоритмів. На базі теорії скінченних автоматів сформовано

багато складних концепцій теоретичної інформатики. Ця теорія має чимало застосувань у технічній інформатиці та становить важливу частину теоретичної інформатики.

Ми розглядатимемо скінченні автомати як абстрактні моделі найпростіших пристроїв опрацювання даних. Спосіб викладення орієнтовано передусім на теорію формальних мов.

Означення формальної мови

Буква (або *символ*) – це простий неподільний знак; множина букв утворює *алфавіт V*. Позаяк алфавіт – це множина, до нього можна застосовувати теоретико-множинні позначення.

Ланцюжки (string) над алфавітом V – це впорядковані сукупності букв алфавіту, і, отже, вони виглядають як елементи множини $V^n = V \times V \times \ldots \times V$. Проте в цьому розділі буде природніше записувати ланцюжки у вигляді $a_1 a_2 \ldots a_n$, а не $(a_1, a_2, \ldots, a_n)$. Букви самі по собі є ланцюжками в разі n=1. Ми будемо допускати випадок, коли ланцюжок не містить букв (*порожній ланцюжок*), і позначатимемо його як λ. Зазначимо, що λ – це не символ, тобто $\lambda \notin V$ для будь-якого алфавіту V.

За аналогією з лінгвістикою ланцюжки іноді називають *словами*. Множину всіх ланцюжків над алфавітом V називають *замиканням* V і позначають V^*, так що

$$V^* = V^0 \cup V^1 \cup V^2 \cup \ldots = \bigcup_{i=0}^{\infty} V^i,$$

де $V^0 = \{\lambda\}$.

Головна операція над ланцюжками – конкатенація. Нехай σ та τ – ланцюжки над алфавітом V. *Конкатенацією* σ та τ називають ланцюжок $\rho = \sigma\tau$ (тобто до ланцюжка σ дописано ланцюжок τ).

Якщо ланцюжок складається з повторюваних символів, то використовують скорочені позначення: для $a \in V$ та цілого невід'ємного n записують

$$a^0 = \lambda;\ aa = a^2;\ aa^{n-1} = a^n.$$

Зазначимо, що є альтернативний набір термінів для букви, алфавіту й слова – відповідно *слово*, *словник* і *речення*. У деяких контекстах ці терміни природніші, проте в цьому випадку особливу увагу потрібно звертати на використання терміна «слово», оскільки він має два значення.

Множину ланцюжків (або речень) називають мовою. Формально *мова* L над алфавітом V – це множина ланцюжків із V^*, тому $L \subset V^*$. Правила, які визначають множину речень, утворюють *синтаксис* мови, а опис множини змістів і відповідності між реченнями та змістами – *семантику* мови. Семантика мови залежить від характеру описуваних нею об'єктів, засоби її вивчення для різних типів мов різні. Семантика мови математики – формальні теорії. Дослідження семантики мов програмування стало окремою частиною теоретичного програмування. Спроби точного опису семантики природних мов стосуються передусім машинного перекладу. Що ж до синтаксису, то його особливості значно менше залежать від призначення мови. Можна сформулювати поняття й методи дослідження синтаксису, які не залежать від змісту та призначення мов. Тому найбільших успіхів математична лінгвістика досягла у вивченні синтаксису, де із середини XX ст. розвинувся спеціальний математичний апарат – теорія формальних породжувальних граматик. Вона дуже важлива теоретично й ефективна в застосуваннях (мовах програмування, штучному інтелекті, машинному перекладі).

Зараз зосередимо увагу на тому, як слова можна складати в речення, і зазначимо, що множина всіх речень, які мають зміст, утворює мову. Нас будуть цікавити здебільшого формальні мови, такі як мови програмування та мови, що описують правильні математичні вирази. Проте спочатку буде корисно навести приклад із природної мови.

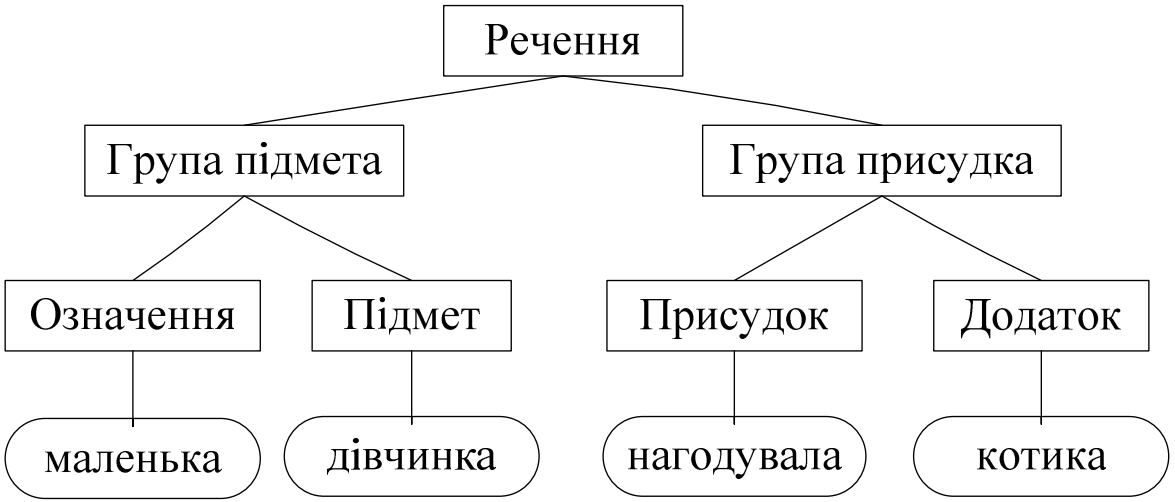

Рис. 1

Приклад 1. Розглянемо речення

«Маленька дівчинка нагодувала котика».

Головним об’єктом нашого розгляду будуть питання синтаксису. Щоб проілюструвати клас структур, які ми вивчатимемо, розглянемо діаграму на рис. 1. Вона означає, що ⟨речення⟩ можна побудувати за допомогою злиття ⟨групи підмета⟩ й ⟨групи присудка⟩, хоча це потребує

формального означення. Група підмета складається з ⟨означення⟩ та ⟨підмета⟩, а група присудка – із ⟨присудка⟩ та ⟨додатка⟩. Остаточно отримаємо ⟨означення⟩ «маленька», ⟨підмет⟩ «дівчинка», ..., що дає нам речення «маленька дівчинка нагодувала котика».

Перш ніж увести термінологію та позначення, потрібні для уточнення загальних понять у конкретній ситуації, яка зображена на рис. 1, зазначимо основні задачі теорії мов.

Нагадаємо, що для заданого алфавіту V мова L – це довільна підмножина множини V^*, проте довільні підмножини становлять незначний інтерес. Ми хочемо зосередити увагу на спеціальних мовах, що містять ланцюжки, які завдяки зовнішній інформації про їх семантику вважають осмисленими чи добре сконструйованими.

Найцікавіші мови нескінченні й, отже, їх неможливо виписати явно. Потрібно придумати способи породження таких мов; як породжувальну систему можна розглядати граматику G. Сформулюємо дві основні задачі формальної теорії мов.

1. Як за заданою граматикою G (і пов'язаною з нею мовою L) породжувати речення $\alpha \in L$?
2. Як за заданими мовою $L \subset V^*$ та ланцюжком $\alpha \in V^*$ з'ясувати, чи $\alpha \in L$?

Щоб перевірити, чи належить деякий ланцюжок (речення) мові L, потрібно знати, як граматика G породжує L. Далі опишемо загальні принципи породжувальних граматик.

Формальні породжувальні граматики

У лінгвістиці природних мов терміни «речення» та «слово» мають різний зміст; тому в математичній лінгвістиці послідовність символів зазвичай називають нейтральним терміном «ланцюжок» (string), а мову, яку розуміють як множину формальних ланцюжків, – формальною мовою.

Формальна породжувальна граматика G (далі – *граматика* G) – це формальна система, визначена четвіркою об'єктів $G = (V, T, S, P)$, де V – скінченна непорожня множина, яку називають *алфавітом* (або *словником*); T – її підмножина, елементи множини T називають *термінальними* (*основними*) символами; S – *початковий символ* ($S \in V$), P – скінченна множина *продукцій* (або *правил перетворення*) вигляду $\xi \to \eta$, де ξ та η – ланцюжки над алфавітом V.

Множину $V \setminus T$ позначають як N, її елементи називають *нетермінальними* (*допоміжними*) символами.

Формальні породжувальні граматики часто називають *граматиками із фразовою структурою* (phrase-structure grammar), *граматиками безпосередніх складових*. Термінальні символи часто називають *терміналами*, а нетермінальні символи – *нетерміналами*.

У теорії формальних граматик усталилися традиції позначень, яких ми будемо дотримуватись. Символи термінального алфавіту позначають малими латинськими буквами чи цифрами, символи нетермінального алфавіту – великими латинськими буквами, ланцюжки над алфавітом V – грецькими буквами. Довжину ланцюжка α позначають $l(\alpha)$ або $|\alpha|$. Нагадаємо, що множину всіх ланцюжків у алфавіті V позначають V^*.

Нас цікавитимуть ланцюжки, які можуть бути породжені продукціями граматики.

Нехай $G = (V, T, S, P)$ – граматика, і нехай $\alpha_0 = \sigma\xi\tau$ (причому $\xi \neq \lambda$), $\alpha_1 = \sigma\eta\tau$ – ланцюжки над V (тут α_0 – конкатенація σ, ξ та τ; α_1 – конкатенація σ, η та τ). Якщо $\xi \to \eta$ – продукція граматики G, то говорять, *що* α_1 *безпосередньо виводиться з* α_0 і записують $\alpha_0 \Rightarrow \alpha_1$.

Якщо $\alpha_0, \alpha_1, \ldots, \alpha_n$ – ланцюжки над V такі, що $\alpha_0 \Rightarrow \alpha_1 \Rightarrow \alpha_2 \Rightarrow \ldots \Rightarrow \alpha_{n-1} \Rightarrow \alpha_n$, то говорять, *що* α_n *виводиться з* α_0, і використовують запис $\alpha_0 \overset{*}{\Rightarrow} \alpha_n$. Послідовність кроків для отримання α_n з α_0 називають *виведенням*.

Приклад 2. Речення українською мовою, наведене в прикладі 1, можна вивести в граматиці $G = (V, T, S, P)$,

де

V={⟨речення⟩, ⟨група підмета⟩, ⟨група присудка⟩, ⟨означення⟩, ⟨підмет⟩, ⟨присудок⟩, ⟨додаток⟩, маленька, дівчинка, нагодувала, котика},

T={маленька, дівчинка, нагодувала, котика},

початковий символ ⟨речення⟩,

а множина P містить такі продукції:

⟨речення⟩ → ⟨група підмета⟩ ⟨група присудка⟩,

⟨група підмета⟩ → ⟨означення⟩ ⟨підмет⟩,

⟨група присудка⟩ → ⟨присудок⟩ ⟨додаток⟩,

⟨означення⟩ → маленька,

⟨підмет⟩ → дівчинка,

⟨присудок⟩ → нагодувала,

⟨додаток⟩ → котика.

Ця система породжує тільки одне речення, тому її можна замінити на

$$L = \{\text{маленька дівчинка нагодувала котика}\}.$$

Якщо ми захочемо розширити мову, щоб увести до неї речення з означеннями «сусідська», «весела», присудком «попестила», і додатками «цуцика», «кролика» (тоді мова L матиме 18 речень), то це можна зробити відповідним розширенням алфавіту та додаванням лише п’яти продукцій до множини P.

Приклад 3. Розглянемо граматику $G = (V, T, S, P)$. Нехай $V = \{a, b, A, B, S\}$, $T = \{a, b\}$, S – початковий символ, $P = \{S \to ABa,\ A \to BB,\ B \to ab,\ AB \to b\}$. Ланцюжок $Aaba$ безпосередньо виводиться з ABa у цій граматиці, оскільки $B \to ab$ – продукція граматики. Ланцюжок $abababa$ виводиться з ланцюжка ABa оскільки

$$ABa \Rightarrow Aaba \Rightarrow BBaba \Rightarrow Bababa \Rightarrow abababa$$

за допомогою послідовного використання продукцій $B \to ab$, $A \to BB$, $B \to ab$ та $B \to ab$.

Нехай $G = (V, T, S, P)$ – граматика. *Мовою, яка породжується G*, називають множину всіх ланцюжків терміналів, які виводяться з початкового символу S. Мову, що породжується граматикою G, позначають $L(G)$. Отже,

$$L(G) = \left\{ \omega \in T^* \mid S \overset{*}{\Rightarrow} \omega \right\},$$

де T^* – множина всіх ланцюжків терміналів, включаючи порожній ланцюжок.

Приклад 4. Нехай G – граматика з алфавітом $V = \{S, A, a, b\}$, множиною терміналів $T = \{a, b\}$, початковим символом S і множиною продукцій $P = \{S \to aA,\ S \to b,\ A \to aa\}$. Знайдемо мову $L(G)$, породжувану цією граматикою.

Із початкового символу S можна вивести ланцюжок aA за допомогою продукції $S \to aA$ чи застосувати продукцію $S \to b$, щоб вивести b. З aA, скориставшись продукцією $A \to aa$, можна вивести ланцюжок aaa. Ніяких інших ланцюжків вивести неможливо. Отже, $L(G) = \{b, aaa\}$.

Приклад 5. Нехай G – граматика з алфавітом $V = \{S, 0, 1\}$, множиною терміналів $T = \{0, 1\}$, початковим символом S і множиною продукцій $P = \{S \to 11S, S \to 0\}$. Знайдемо $L(G)$.

Із початкового символу S одержимо 0 (за допомогою другої продукції) або $11S$ (за допомогою першої). Із ланцюжка $11S$ можна отримати 110 або $1111S$, із ланцюжка $1111S$ виводяться 11110 або $111111S$. Отже, після кожного виведення ми або додаємо дві одиниці в кінець ланцюжка, або закінчуємо ланцюжок нулем. Звідси випливає, що $L(G) = \{0, 110, 11110, 1111110, \ldots\}$ – це множина всіх ланцюжків з парною кількістю тільки 1, після яких (у кінці) обов'язково один 0.

Зауваження. Ми отримали нескінченну мову (тому що $L(G)$ складається з нескінченної кількості ланцюжків). Щоб граматика G породжувала нескінченну мову, у множині продукцій має бути принаймні одне рекурсивне правило (у прикладі 5 це правило $S \to 11S$).

Важливою є проблема побудови граматики для заданої мови.

Приклад 6. Знайдемо граматику, яка породжує мову $\left\{0^m 1^m \,\middle|\, m = 0,1,2,\ldots\right\}$. Потрібно дві продукції, щоб побудувати ланцюжок із якоїсь кількості 0, після яких є така сама кількість 1. Перша продукція збільшує ланцюжок зсередини, додаючи зліва від центру 0, а справа – 1. Друга продукція замінює початковий символ S на порожній ланцюжок λ. Одержимо граматику $G = (V, T, S, P)$, де $V = \{0, 1, S\}$, $T = \{0, 1\}$, S – початковий символ, $P = \{S \to 0S1, S \to \lambda\}$.

Приклад 7. Знайти граматику, яка породжує мову $\left\{0^m 1^n \,\middle|\, m,n = 0,1,2,\ldots\right\}$.

Наведемо дві такі граматики:

граматика G_1:

$$V=\{S, 0, 1\},\ T=\{0, 1\},\ P=\{S\to 0S,\ S\to S1,\ S\to\lambda\};$$

та граматика G_2:

$$V=\{S, A, 0, 1\},\ T=\{0, 1\},\ P=\{S\to 0S,\ S\to 1A,\ S\to 1,\ A\to 1A,\ A\to 1,\ S\to\lambda\}.$$

Пропонуємо обґрунтувати, що обидві граматики справді породжують мову $\left\{0^m 1^n \,\middle|\, m,n = 0,1,2,\ldots\right\}$. Цей приклад свідчить, що дві різні граматики можуть породжувати одну мову.

Граматики G_1 і G_2 називають *еквівалентними*, якщо $L(G_1) = L(G_2)$. Граматики G_1 та G_2 із прикладу 7 еквівалентні.

Іноді мову, яку легко описати, доводиться задавати досить складною граматикою.

Приклад 8. Побудуємо граматику, яка породжує мову $\left\{0^m 1^m 2^m \,\middle|\, m = 0,1,2,\ldots\right\}$.

Розв'язок цієї задачі – така граматика: $G = (V, T, S, P)$, де $V = \{0, 1, 2, S, A, B\}$, $T=\{0, 1, 2\}$, початковий символ S, множина продукцій

$P = \{S\to 0SAB,\ S\to\lambda,\ BA\to AB,\ 0A\to 01,\ 1A\to 11,\ 1B\to 12,\ 2B\to 22\}$.

Типи граматик (ієрархія Хомського)

Продукція граматики дає змогу заміняти одну послідовність символів іншою. Граматики класифікують за типами продукцій. Розглянемо класифікацію, яку запропонував американський математик і лінгвіст Н. Хомський (N. Chomsky).

Тип граматики	Обмеження на продукції $\xi \to \eta$
0	Немає обмежень
1	$\lvert\xi\rvert \le \lvert\eta\rvert$, або $\eta = \lambda$
2	$\xi = A$, де A – нетермінальний символ
3	$\xi = A$, причому $\eta = aB$ або $\eta = a$, де A, B – нетермінальні символи, a – термінальний символ, або $S \to \lambda$

Принцип цієї класифікації полягає в тому, що на продукції накладено певні обмеження.

Граматика *типу 0* не має жодних обмежень на свої продукції. Граматика *типу 1* може мати продукції лише у вигляді $\xi \to \eta$, де довжина ланцюжка η не менша, ніж довжина ланцюжка ξ, або у вигляді $\xi \to \lambda$. У граматиці *типу 2* можуть бути лише продукції у вигляді $A \to \eta$, де A – нетермінальний символ. Граматика *типу 3* може мати лише такі продукції: $A \to aB$, $A \to a$, $S \to \lambda$, де A, B – нетермінали, a – термінал.

Із цих означень випливає, що кожна граматика типу 3 є граматикою типу 2, кожна граматика типу 2 є граматикою типу 1, і, нарешті, кожна граматика типу 1 є граматикою типу 0.

Граматики типу 2 називають *контекстно вільними*, бо нетермінал A у лівій частині продукції $A\to\eta$ може бути замінений ланцюжком η у довільному оточенні щоразу, коли він зустрічається, тобто незалежно від контексту. Мову, яку породжує граматика типу 2, називають *контекстно вільною*.

Граматику типу 1 називають *контекстно залежною*. Якщо в множині продукцій P є продукція вигляду $\gamma\mu\delta\to\gamma\nu\delta$, $|\mu|\le|\nu|$ (але не у формі $\mu\to\nu$), то μ можна замінити на ν лише в оточенні ланцюжків $\gamma\ldots\delta$, тобто у відповідному контексті. Мову, яку породжує граматика типу 1, називають *контекстно залежною*.

Граматику типу 3 називають *регулярною*. Нагадаємо, що в ній можуть бути лише продукції $A\to aB$, $A\to a$, $S\to\lambda$, де A, B – нетермінали, a – термінал. Мову, яку породжує граматика типу 3, називають *регулярною*.

Приклад 9. Мова $\left\{0^m1^n \middle| m,n=0,1,2,\ldots\right\}$ регулярна, бо її породжує регулярна граматика G_2 прикладу 7.

Приклад 10. Мова $\left\{0^m 1^m \middle| m = 0,1,2,\ldots\right\}$ контекстно вільна, оскільки вона породжена граматикою із продукціями $S \to 0S1$ та $S \to \lambda$ (див. прикл. 6). Проте ця мова не регулярна: не існує регулярної граматики, яка породжувала б цю мову. Цей факт потребує окремого доведення.

Приклад 11. Мова $\left\{0^m 1^m 2^m \middle| m = 0,1,2,\ldots\right\}$ контекстно залежна, тому що вона може бути породжена граматикою типу 1 (див. прикл. 8). Проте вона не може бути породжена жодною граматикою типу 2; це треба довести окремо.

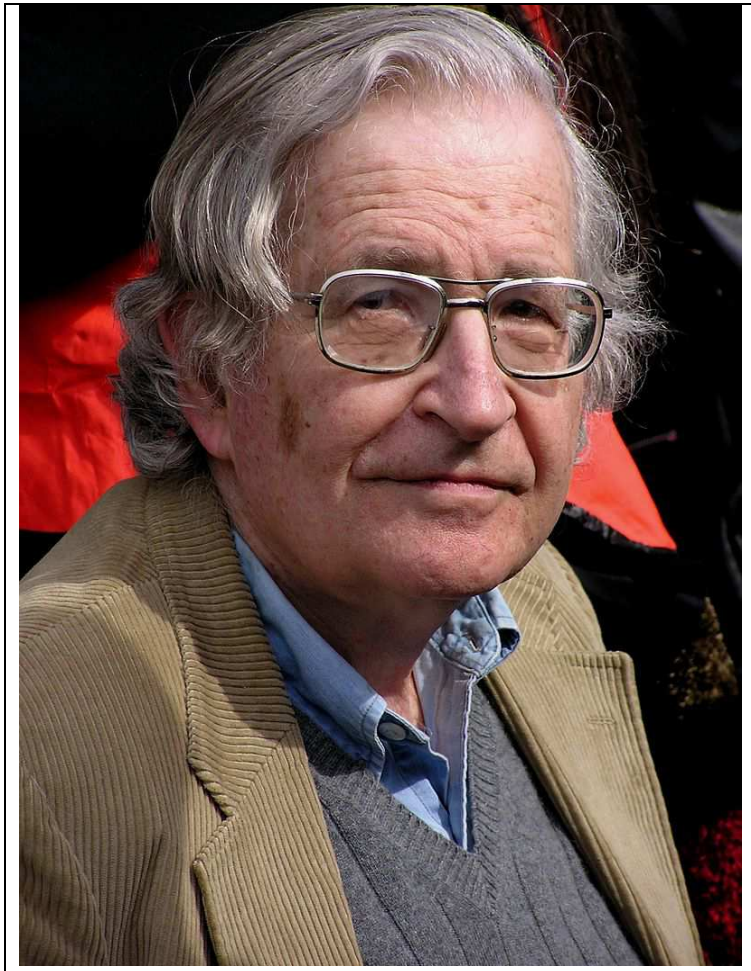

Фото 2004 р.

Ноа́м Авра́м Хо́мський (також транскрибується як *Чо́мскі*, англ. *Noam Avram Chomsky*)
– американський лінгвіст, філософ та політичний активіст, аналітик, літератор, професор мовознавства Масачусетського технологічного інституту (MTI) у відставці. Хомський добре відомий академічній та науковій спільноті як один із засновників сучасної лінгвістики та визначна постать в аналітичній філософії.
Хомський є автором понад 100 книг.
Нагороди: 1988 – Кіотська премія (аналог Нобелівської премії); 1999 – медаль Б. Франкліна (США).
Ноам Аврам Хомський народився 7 грудня 1928 року в Філадельфії, штат Пенсільванія, в єврейській родині. Його батько — знаний професор івриту Вільям Хомський, був родом з України, мати — Еліза Сімоновська, мала білоруські корені. (*Н. Хомський*: *My father came from the Ukraine... —* «*Мій батько походить з України...*» — з інтерв'ю Н.Хомського).

Дерева виведення

У мовах, породжених контекстно вільними граматиками, виведення можна зображати графічно за допомогою орієнтованих кореневих дерев. Їх називають *деревами виведення*, або *деревами синтаксичного розбору*. Кореню дерева виведення відповідає початковий символ, внутрішнім вершинам – нетермінальні символи, що зустрічаються у виведенні, листкам – термінальні символи. Нехай γ – ланцюжок і $A\to\gamma$ – продукція, використана у виведенні. Тоді вершина, яка відповідає нетермінальному символу A, має синами вершини, які відповідають кожному символу ланцюжка γ у порядку зліва направо.

Приклад 12. Визначимо, чи ланцюжок *cbab* належить мові, породженій граматикою $G=(V, T, S, P)$, де $V=\{a, b, c, A, B, C, S\}$, $T=\{a, b, c\}$, S – початковий символ, а множина продукцій

$$P=\{S\to AB,\ A\to Ca,\ B\to Ba,\ B\to Cb,\ B\to b,\ C\to cb,\ C\to b\}.$$

Розв'язати цю задачу можна двома способами.

Розбір зверху вниз. Оскільки є лише одна продукція з початковим символом S у лівій частині, то виведення починаємо з $S\Rightarrow AB$. Далі використаємо продукцію $A\to Ca$. Отже, маємо $S\Rightarrow AB\Rightarrow CaB$. Позаяк ланцюжок *cbab* починається із символів *cb*, то, використавши продукцію $C\to cb$, одержимо $S\Rightarrow AB\Rightarrow CaB\Rightarrow cbaB$. Завершуємо виведення використанням продукції $B\to b$:

$$S\Rightarrow AB\Rightarrow CaB\Rightarrow cbaB\Rightarrow cbab.$$

Отже, ланцюжок *cbab* належить мові $L(G)$.

Розбір знизу вверх. Почнемо з ланцюжка *cbab*, який потрібно вивести. Можна використати продукцію $C\to cb$, отже, $Cab\Rightarrow cbab$. Застосувавши продукцію $A\to Ca$, отримаємо $Ab\Rightarrow Cab\Rightarrow cbab$. Тепер використаємо продукцію $B\to b$, одержимо $AB\Rightarrow Ab\Rightarrow Cab\Rightarrow cbab$. Нарешті, застосуємо продукцію $S\to AB$:

$$S\Rightarrow AB\Rightarrow Ab\Rightarrow Cab\Rightarrow cbab.$$

Дерево виведення для рядка *cbab* у граматиці *G* зображено на рис. 2.

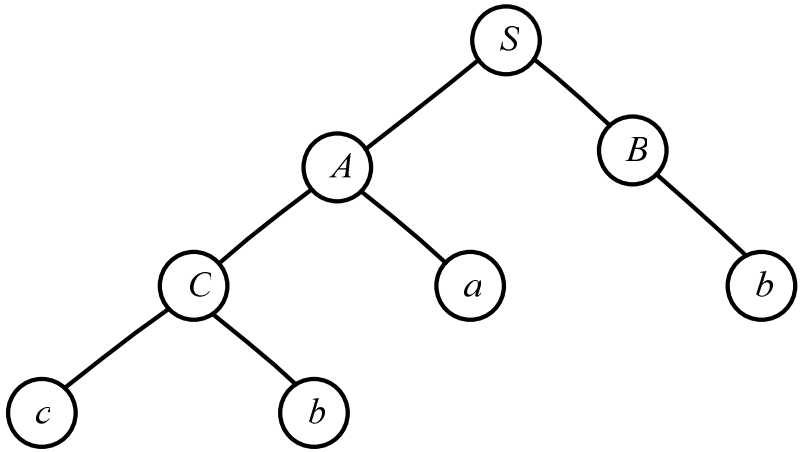

Рис. 2

Виведення називають *еквівалентними*, якщо їм відповідають однакові дерева. Отже, у прикладі 12 ми отримали еквівалентні виведення. Перевага дерева виведення порівняно з виведенням – його компактність. Дерево на рис. 2 має вісім вершин, тоді як відповідні виведення – 18 або 16 символів.

Взаємно однозначної відповідності між ланцюжками мови *L* і деревами виведення в граматиці *G*, яка породжує *L*, може й не бути. Контекстно вільну граматику *G* називають *неоднозначною*, якщо існує хоча б один ланцюжок у мові *L*(*G*), який має в *G* більше одного дерева виведення.

Форми Бекуса–Наура

Для граматик типу 2 (контекстно вільних), окрім звичайного, є й інший спосіб подання – *форми Бекуса–Наура*. У лівій частині продукцій граматик типу 2 один символ (нетермінальний). Замість того щоб виписувати окремо всі продукції, можна об'єднати в один вираз продукції з однаковими символами в лівій частині. Тоді замість символу → в продукціях використовують символ ::=. Усі нетермінали в цьому разі беруть у трикутні дужки ⟨ ⟩. Праві частини продукцій в одному виразі відокремлюють одну від одної символом |.

Приклад 13. Три продукції $A \to Aa$, $A \to a$ та $A \to AB$ можна подати одним таким виразом у формі Бекуса–Наура: ⟨*A*⟩::=⟨*A*⟩*a*|*a*|⟨*A*⟩⟨*B*⟩.

Приклад 14. Знайдемо продукції граматики, якщо у формі Бекуса–Наура їх записано так:

⟨*expression*⟩::=(⟨*expression*⟩)|⟨*expression*⟩+
⟨*expression*⟩|⟨*expression*⟩*⟨*expression*⟩| ⟨*variable*⟩;
⟨*variable*⟩::=*x*|*y*.

Побудуємо дерево виведення в цій граматиці для ланцюжка $(x*y)+x$.

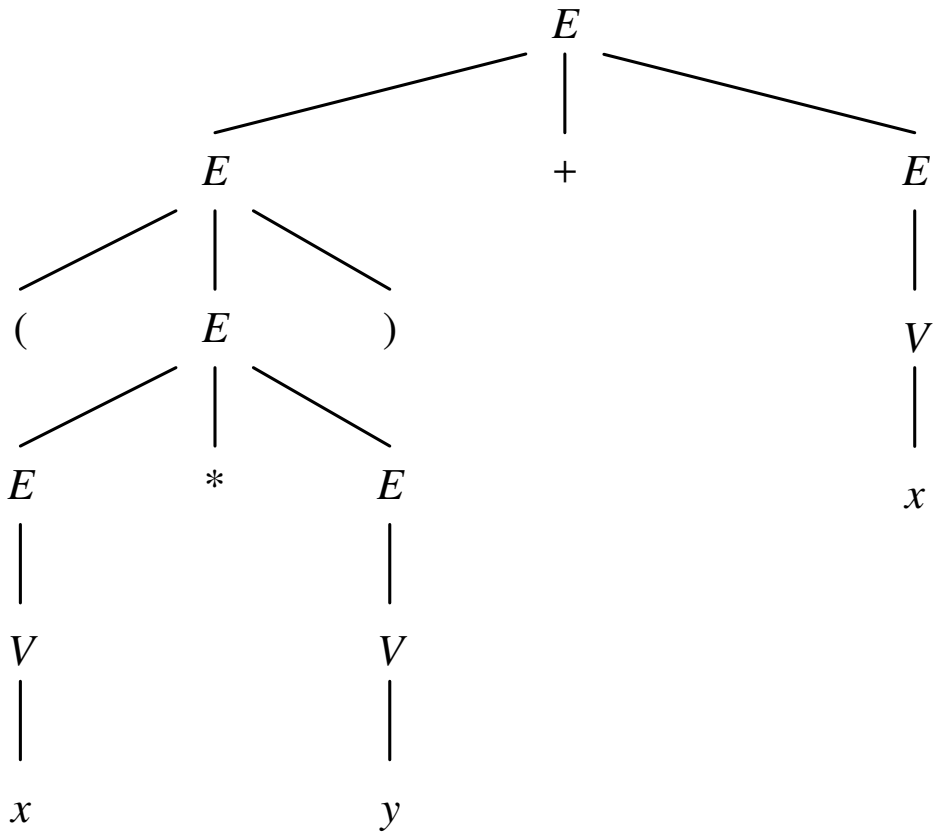

Рис. 3

Для зручності використаємо позначення E для ⟨*expression*⟩ (це буде також і початковий символ) та V для ⟨*variable*⟩. Тоді правил перетворення (продукції граматики) – $E\to(E)$, $E\to E+E$, $E\to E*E$ та $E\to V$ з першого виразу, а також $V\to x$ та $V\to y$ із другого.

Дерево виведення для ланцюжка $(x*y)+x$ зображено на рис. 3.