## 6.3 Розпізнавачі для різних класів граматик

Під розпізнавачем будемо розуміти узагальнений алгоритм, що дозволяє визначити деяку множину (в нашому випадку – мову) і використовує в своїй роботі такі компоненти: вхідну стрічку, керуючий пристрій з кінцевої пам'яттю і додаткову робочу пам'ять. Як приклади розпізнавачів можна назвати машину Тьюринга, скінченні та магазинні автомати.

Виявляється, кожному класу граматик з ієрархії Хомського відповідає клас розпізнавачів, що визначає той же клас мов:

– мова L регулярна тоді і тільки тоді, коли вона визначається (одностороннім детермінованим) кінцевим автоматом;

– мова L контекстно-вільна тоді і тільки тоді, коли вона визначається (одностороннім недетермінованим) автоматом з магазинною пам'яттю;

– мова L контекстно-залежна тоді і тільки тоді, коли вона визначається (двостороннім недетермінованим) автоматом з магазинною пам'яттю;

– мова L рекурсивно-перелічувальна тоді і тільки тоді, коли вона визначається машиною Тьюринга.

Розпізнавач:

Деякий алгоритм, що дозволяє визначити мову (L) і використовує:

1) вхідну стрічку;

2) керуючий пристрій з скінченою пам'яттю;

3) доповнювач до робочої пам'яті (машина Тьюринга, кінцеві автомати, магазинний автомат).

Якщо розпізнавач зупиняється в заключному стані, то ланцюжок належить мові (див. табл. 1).

**Таблиця 1**

| Тип | Мови за Хомським | Розпізнавач |
|---|---|---|
| 3 | Регулярна мова | Скінчений автомат (односторонній детермінований) |
| 2 | Контекстно-вільна мова | Автомат з магазинною пам’яттю (односторонній недетермінований) |
| 1 | Контекстно-залежна мова | Лінійно-обмежений автомат (двосторонній недетермінований) |
| 0 | Рекурсивно-перелічувальна мова | Машина Тьюринга |

## Способи визначення формальних мов (продовження)

Відомі різні способи опису мов. Скінченну мову можна описати простим перерахуванням її ланцюжків. Оскільки формальна мова може бути і нескінченною, потрібні механізми, що дозволяють скінченним чином представляти нескінченні мови. Можна виділити два основних підходи для такого представлення: механізм розпізнавання і механізм породження (генерації).

Основний спосіб реалізації *механізму породження* – використання граматик, що породжують, які також називають граматиками Хомського. Граматики, що породжують, було розглянуто минулої лекції.

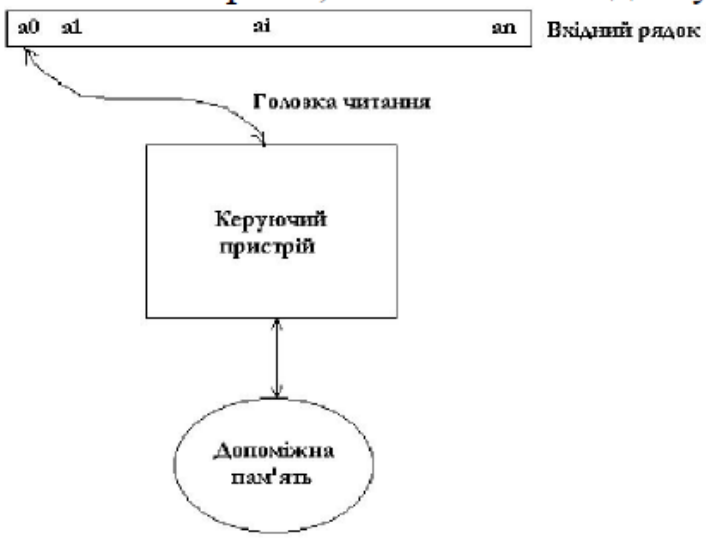
*Механізм розпізнавання* (*розпізнавач*), по суті, є процедурою спеціального виду, яка за заданим ланцюжком визначає, чи належить він мові. Якщо належить, то процедура зупиняється з відповіддю «так», тобто допускає ланцюжок; інакше – зупиняється з відповіддю «ні» або зациклюється. Мова, яка визначається розпізнавачем – це безліч всіх ланцюжків, які він допускає.

Приклади розпізнавачів:

- Машина Тьюринга (МТ). Мова, яку можна задати за допомогою МТ, називається рекурсивно-перелічуваною. Замість МТ можна використовувати еквівалентні алгоритмічні схеми: нормальний алгоритм Маркова (НАМ), машину Поста та ін.
- Лінійно-обмежений автомат (ЛОА). Являє собою МТ, в якій стрічка не нескінченна, а обмежена довжиною вхідного слова. Визначає контекстно-залежні мови.
- Автомат з магазинної (зовнішньої) пам'яттю (МП-автомат). На відміну від ЛОА, головка не може змінювати вхідне слово і не може зрушуватися вліво; є додаткова нескінченна пам'ять (магазин, або стек). Визначає контекстно-вільні мови.
- Скінченний автомат (СА). Відрізняється від МП-автомата відсутністю магазину. Визначає регулярні мови.

## *Розпізнавачі*

***Розпізнавач*** – це схематизований алгоритм, який визначає деяку множину ланцюжків. Він складається з трьох частин:




- вхідна стрічка;
- керуючий пристрій із скінченною пам'яттю;
- робоча (допоміжна) пам'ять.

***Вхідна стрічка*** – це послідовність комірок, в якій можуть знаходитись символи алфавіту. У кожен момент часу головка читання переглядає одну комірку, За один крок головка може зсунутись на одну комірку вправо, вліво або ж залишитися на місці. Розпізнавач, який ніколи не зсуває головку вліво називається ***одностороннім***. Як правило вважається, що вхідна головка читає, тобто під час роботи розпізнається якийсь символ.

***Пам'ять розпізнавача*** – сховище інформації або даних. Внутрішній алфавіт пам'яті $Q$ скінченний і інформація в пам'яті побудована тільки із символів даного алфавіту. Описати пам'ять (її вміст і структуру) можна скінченними засобами, а об'єм пам'яті може бути як завгодно великим. Важливим прикладом допоміжної пам'яті є магазинна пам'ять. Магазинна пам'ять містить ланцюжок символів $z_1 z_2 ... z_n, z_i \in Q$, а $z_1$- верхній символ магазину. Поведінка допоміжної пам'яті характеризується за допомогою двох функцій: функція доступу і функція перетворення пам'яті.

***Функція доступу до пам'яті*** відображає множину можливих станів (конфігурацій) у можливу множину інформаційних символів. Функція доступу до пам'яті у випадку магазинної пам'яті визначається таким чином: $f(z_1, z_2, ..., z_n) = z_1$.

***Функція перетворення пам'яті*** – це правило, яке описує зміну пам'яті. Тип допоміжної пам'яті визначає назву розпізнавача. Якщо пам'ять влаштована, як магазин, то розпізнавач називається розпізнавачем із магазинною пам'яттю.

***Ядром розпізнавача*** є керуючий пристрій із скінченною пам'яттю, під яким можна розуміти програму, що керує роботою розпізнавача. Керуючий пристрій являє собою скінченну множину станів. Він керує рухом головки читання. Розпізнавач працює дискретно, виконуючи послідовно такти. На початку такту зчитується поточний символ і за допомогою функції доступу досліджується пам'ять. Вхідний символ, інформація із пам'яті і поточний стан керуючого пристрою визначають дії під час такту:

- вхідна головка рухається вліво, вправо або залишається на місці;
- в пам'яті розміщується деяка інформація і змінюється стан керуючого пристрою.

Поведінку розпізнавача зручно задавати у термінах конфігурацій. У кожен момент часу ***конфігурація*** містить інформацію про

- стан керуючого пристрою;
- містиме вхідної стрічки разом з положенням вхідної головки;
- містиме пам'яті.

Керуючий пристрій може бути детермінованим і недетермінованим. Якщо керуючий пристрій недетермінований, то для довільної конфігурації існує скінченна кількість наступних кроків. Будь-який з них розпізнавач може зробити.

Керуючий пристрій називається ***детермінованим***, якщо для кожної конфігурації існує не більше одного наступного детермінованого кроку.

Конфігурація називається ***початковою***, якщо керуючий пристрій знаходиться в заданому початковому стані, вхідна головка розглядає найлівіший символ, а пам'ять має початкове містиме.

Конфігурація називається ***заключною***, якщо керуючий пристрій знаходиться в одному із заключних станів, а вхідна головка оглядає правий кінець стрічки. Іноді вимагається, щоб в заключній конфігурації і пам'ять задовольняла певним умовам.

Кажуть, що розпізнавач ***допускає*** вхідний рядок $\omega$, якщо починаючи із початкової конфігурації, розпізнавач може виконати послідовність кроків, що закінчиться заключною конфігурацією.

Якщо розпізнавач недетермінований, то він може виконувати різну послідовність кроків. Якщо одна з цих послідовностей закінчується заключною конфігурацією, то вхідний рядок ***допустимий***.

Мова, що дозволяється розпізнавачем – це множина вхідних ланцюжків, які він допускає.

Для кожного класу граматик із ієрархії Хомського існує свій клас розпізнавачів, які визначають ті самі класи мов, що і граматики, наприклад:

- мова L праволінійна тоді і тільки тоді, коли вона розпізнається одностороннім детермінованим скінченним розпізнавачем;
- мова L контекстно-вільна тоді і тільки тоді, коли вона визначається одностороннім недетермінованим розпізнавачем з магазинною пам'яттю;
- мова L контекстно-залежна тоді і тільки тоді, коли вона розпізнається двостороннім недетермінованим обмеженим розпізнавачем.

### *Скінченні автомати*

Неформально скінченний автомат можна розуміти як систему, яка у кожен момент часу може знаходиться в одному з скінченної кількості заданих станів. Кожен такт автомата полягає в тому, що при перебуванні у будь-якому стані і зчитуванні одного з вхідних символів він переходить у визначений стан, причому символи, які зчитувались раніше не запам'ятовуються. Скінчений автомат – це односторонній скінченний розпізнавач без пам'яті.

Формально ***скінченним автоматом*** M називається п'ятірка позначень $M = (Q, \Sigma, \delta, q_0, F)$, де

$Q$ - скінченна множина станів автомату $Q = \{q_0, q_1, \ldots, q_n\}$;

$\Sigma$ - скінченна множина вхідних символів $\Sigma = \{a_1, a_2, \ldots, a_m\}$;

$\delta$ - відображення множини $Q \times \Sigma$ в $P(Q)$ (множину всіх підмножин $Q$), що визначає поведінку керуючого пристрою (функцію $\delta(q, a)$, де $q \in Q$, $a \in \Sigma$, називають функцією переходів);

$q_0 \in Q$ - початковий стан керуючого пристрою;

$F \subseteq Q$ - множина заключних станів.

Робота скінченного автомату полягає у виконанні послідовності кроків (тактів). Такт визначається поточним станом керуючого пристрою, який оглядається головкою читання. Зміна стану автомата – це зміна керуючого пристрою і зсув головки вправо на одну комірку.


Для того, щоб визначити поведінку скінченного автомата треба знати

- поточний стан керуючого пристрою;
- ланцюжок символів, що починаються з комірки, на який вказує головка читання.

Ці два параметри визначають миттєвий опис скінченного автомата, який називається ***конфігурацією (тактом)***.

Пара $(q,\omega) \in Q \times \Sigma^*$ називається ***конфігурацією*** автомата $M$.

Конфігурація $(q_0,\omega)$ називається ***початковою конфігурацією***, а $(q,e)$, $q \in F$ - ***заключною***.

Кажуть, що автомат переходить з конфігурації $(q, a\omega)$, $a \in \Sigma$, в конфігурацію $(q', \omega)$, якщо $q' \in \delta(q,a)$ і позначають це так: $(q,a\omega) \mapsto (q',\omega)$.

Робота скінченого автомата – це послідовність конфігурацій. Нехай $K_0, K_1, ..., K_p$ - деякі конфігурації автомата. Якщо можна перейти $K_0 \mapsto K_1 \mapsto ... \mapsto K_p$, то це можна позначити так: $K_0 \overset{*}{\mapsto} K_p$ або $K_0 \overset{p}{\mapsto} K_p$.

Кажуть, що автомат $M$ ***допускає ланцюжок*** $\omega$, якщо $(q_0,\omega) \overset{p}{\mapsto} (q,e)$, де $q \in F$.

***Мовою***, що визначається (допускається) автоматом $M$, називається множина вхідних ланцюжків, що допускається цим автоматом.

$$L(M) = \{\omega \mid \omega \in \Sigma^*, (q_0,\omega\} \overset{*}{\mapsto} (q,e) \text{ для деякого } q \in F\}.$$

Стан $p$ називається ***досяжним***, якщо існує ланцюжок $\omega$ такий, що $(q_0,\omega) \overset{*}{\mapsto} (p,e)$.

За даним скінченним автоматом можна знайти найменший еквівалентний йому автомат, якщо вилучити всі недосяжні стани, а потім склеїти еквівалентні стани.

Кажуть, що ланцюжок $x \in \Sigma^*$ ***розрізняє*** стани $q_1 \neq q_2$, $q_1 \in Q$, $q_2 \in Q$, якщо із стану $q_1$ по ланцюжку $x$ можна перейти в $q_3$, а з

$q_2$ по $x$ можна перейти в $q_4$, причому $q_3 \in F$, $q_4 \notin F$ або $q_3 \notin F$, $q_4 \in F$.

Якщо ланцюжок $x$ такий, що $(q_1, x) \mapsto^k (q_3, e)$, $(q_2, x) \mapsto^k (q_4, e)$, то ланцюжок $x$ довжиною $k$ розрізняє ці стани.

Кажуть, що стани $q_1, q_2$ $k-$***не розрізняються:*** $q_1 \equiv^k q_2$, якщо не існує ланцюжка $x \in \Sigma^*, |x| \le k.$ , який розрізняє ці стани.

Будемо говорити, що $q_1, q_2$ ***не розрізняються*** і писати $q_1 \equiv q_2$, якщо вони $k$ -не розрізняються для будь-якого $k \in N$.

Стан $q$ називається ***недосяжним***, якщо не існує вхідного ланцюжка $x \in \Sigma^* : (q_0, x) \mapsto^* (q, e)$.

Скінченний автомат $M$ є ***недетермінованим***, якщо існують такі $a \in \Sigma, q \in Q$, що функція переходів $\delta$ є множиною, що складається більш, ніж з одного стану: $\delta(q,a) = \{q_1, q_2, ..., q_n\}$, $n > 1$.

Скінченний автомат $M$ є ***детермінованим***, якщо для кожних $a \in \Sigma, q \in Q$ існує не більше одного стану, в який відбувається перехід.

Скінченний детермінований автомат називається ***повністю визначеним***, якщо $\delta(q,a)$ для кожних $a \in \Sigma, q \in Q$ містить рівно один стан.

Скінченний детермінований автомат $M$ називається ***зведеним (мінімальним)***, якщо в $Q$ немає недосяжних станів і немає двох станів, що не розрізняються.

***Діаграмою переходів*** (графом переходів) автомата $M$ називається навантажений граф, вершини якого навантажені іменами станів автомату, і в діаграмі присутнє ребро $(p, q)$, якщо справджується таке співвідношення: $q \in \delta(p, a), a \in \Sigma$. Ребра навантажені всіма символами $a \in \Sigma$, по яких є перехід з $p$ в $q$.

Тобто, діаграмою переходів скінченого автомата є граф, вершинами якого є стани автомата, а ребра графа визначаються функцією переходу автомата.

**Приклад недетермінованого скінченного автомата**

Побудуємо автомат в алфавіті $\Sigma = \{1,2,3\}$, який буде розпізнавати слова, в яких є хоча б одна двійка.

Наприклад, $w_1 = 12231 \in L(M)$, $w_2 = 1131131 \notin L(M)$.
Визначимо стани:

$p$ - початковий стан: двійки ще не було

$q$ - заключний стан: двійка вже була.

Отже, автомат має вигляд $M = (\{p,q\},\{1,2,3\},\delta,p,\{q\})$.

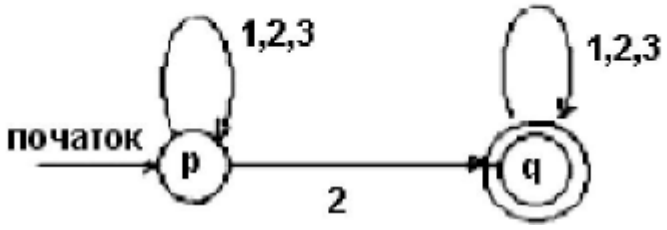
Визначимо функцію переходів $\delta$:

$$\delta(p,1) = p, \delta(p,2) = \{p,q\}, \delta(p,3) = p,$$
$$\delta(q,1) = q, \delta(q,2) = q, \delta(q,3) = q.$$

Функцію $\delta$ прийнято зображати у вигляді ***таблиці переходів***:

| Стани | Вхідні символи | | |
|---|---|---|---|
| | 1 | 2 | 3 |
| $p$ | $\{p\}$ | $\{p,q\}$ | $\{p\}$ |
| (q) | $\{q\}$ | $\{q\}$ | $\{q\}$ |

Функцію $\delta$ можна зобразити у вигляді ***діаграми переходів***:




Заключний стан у таблиці обведений, а на діаграмі – знаходиться у подвійному кружечку.

Нехай у нас є ланцюжок 12231. Випишемо всі можливі послідовності конфігурацій:

$(p,12231) \mapsto (p,2231) \mapsto (p,231) \mapsto (p,31) \mapsto (p,1) \mapsto (p,\text{e})$
$(p,12231) \mapsto (p,2231) \mapsto (p,231) \mapsto (q,31) \mapsto (q,1) \mapsto (q,\text{e})$

$(p,12231) \mapsto (p,2231) \mapsto (q,231) \mapsto (q,31) \mapsto (q,1) \mapsto (q,\text{e})$

Оскільки серед останніх конфігурацій присутня заключна конфігурація $(q,\text{e})$, то існує послідовність, що перетворює початкову конфігурацію $(p,12231)$ в заключну. Це означає, що ланцюжок $12231 \in L(M)$.

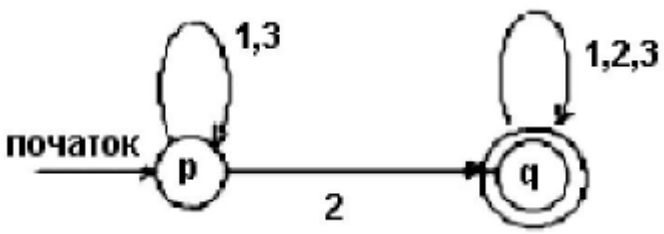
**Приклад детермінованого скінченного автомата**

Знов побудуємо автомат в алфавіті $\Sigma = \{1,2,3\}$, який буде розпізнавати слова, в яких є хоча б одна двійка. Але тепер він буде детермінованим.

Визначимо такі ж стани, як і раніше, але змінимо функцію переходів $\delta$:

$$\delta(p,1) = p, \delta(p,2) = \{q\}, \delta(p,3) = p,$$
$$\delta(q,1) = q, \delta(q,2) = q, \delta(q,3) = q.$$

| Стани | Вхідні символи | | |
|---|---|---|---|
| | 1 | 2 | 3 |
| $p$ | $\{p\}$ | $\{q\}$ | $\{p\}$ |
| $(q)$ | $\{q\}$ | $\{q\}$ | $\{q\}$ |



Отже, автомат має вигляд $M = (\{p,q\},\{1,2,3\},\delta,p,\{q\})$. Він повністю визначений і мінімальний.

Оскільки автомат детермінований, то для будь-якого ланцюжка існує єдина послідовність конфігурацій. Нехай у нас є ланцюжок 12231. Тоді послідовність конфігурацій має вигляд:

$$(p,12231) \mapsto (p,2231) \mapsto (q,231) \mapsto (q,31) \mapsto (q,1) \mapsto (q,\text{e})$$

Ця послідовність перетворює початкову конфігурацію в заключну. Це означає, що ланцюжок $12231 \in L(M)$.

Зазначимо, що працювати з детермінованим автоматом простіше, ніж з недермінованим, але, як правило, для складних мов його непросто визначити. Існують алгоритми перетворення недетермінованого автомата в детермінований, а потім – і у мінімальний. Вони розглянуті далі, а нижче наведено алгоритм моделювання роботи скінченного детермінованого автомату:

```
q = q0;                 // початковий стан автомата
a = GetChar();          // зчитування першого символу з вхідної стрічки
while (a != eof)        // поки не кінець стрічки
{
  // перехід автомату в інший стан по прочитаному символу
  q = Delta(q, a);      // функція переходів Delta керує роботою автомата
  a = GetChar();        // зчитування наступного символу з вхідної стрічки
}                       // стрічку прочитано
if (q is in F)          // автомат попав в заключний стан ?
       return "yes";    // ланцюжок належить мові автомата
else return "no";       // ланцюжок не належить мові автомата
```

## *Магазинні автомати*

Магазинні автомати, відомі також як автомати з магазинною пам'яттю або як МП-автомати, формально визначаються у такий спосіб:

МП-автомат – це сімка позначень P = (Q, Σ, Г, δ, $q_0$, $Z_0$, F), де:

Q – скінченна множина станів;

Σ – скінченний вхідний алфавіт;

Г – скінченний алфавіт магазинних символів;

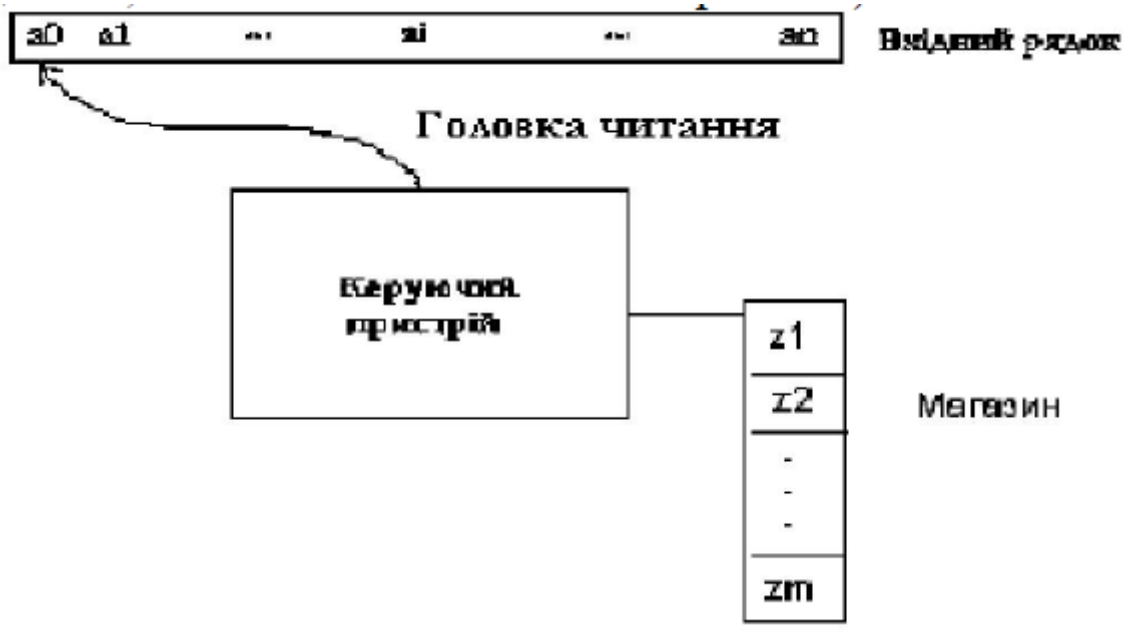
δ – функція переходу, відображення множини $Q \times (\Sigma \cup \{e\}) \times \Gamma$ у множину скінченних підмножин множини $Q \times \Gamma^*$;

$q_0 \in Q$ – початковий стан керуючого пристрою;

$Z_0 \in \Gamma$ – символ, що знаходиться в магазині у початковий момент (початковий символ);

$F \subseteq Q$ – множина заключних станів.

***Конфігурацією*** МП-автомата $P$ називається трійка позначень $(q,\omega,\alpha) \in Q \times \Sigma^* \times \Gamma^*$, де $q$ – поточний стан керуючого пристрою, $\omega$ – невикористана частина вхідного ланцюжка (якщо $\omega=\varepsilon$, то вважається, що весь вхідний ланцюжок прочитаний), $\alpha$ – вміст магазина (самий лівий символ ланцюжка $\alpha$ вважається верхнім символом магазина; якщо $\alpha=\varepsilon$, те магазин вважається порожнім).



На кожному кроці роботи МП-автомат може або занести щось у магазин, або зняти якісь значення з його вершини. Відзначимо, що МП-автомат може продовжувати працювати у випадку закінчення вхідного ланцюжка, але не може продовжувати роботу у випадку спустошення магазина.

Клас мов, що розпізнаються МП-автоматами, у точності збігається з класом мов, які задаються контекстно-вільними граматиками.

МП-автомат P = (Q, Σ, Γ, δ, $q_0$, $Z_0$, F) називається ***детермінованим***, якщо для кожних $q \in Q$ і $Z \in \Gamma$ вірно одне з наступних тверджень:

- δ($q$, $a$, $Z$) містить не більш одного елемента для кожного $a \in \Sigma$ і δ($q$, $\varepsilon$, $Z$) = Ø;
- δ($q$, $a$, $Z$) = Ø для всіх $a \in \Sigma$ і δ($q$, $\varepsilon$, $Z$) містить не більш одного елемента.

Детерміновані МП-автомати описують тільки підмножину з класу контекстно-вільних мов – ця підмножина називається ***детермінованими контекстно-вільними мовами***. Цей клас мов називають також LR(k)-граматиками, тому що вони можуть бути однозначно розібрані шляхом перегляду ланцюжка зліва направо із загляданням уперед не більш, ніж на $k$ символів.

**Приклад магазинного автомата**

Розглянемо магазинний автомат, що розпізнає мову $\{0^n1^n \mid n \geq 0\}$. Нехай P = ($\{q_0, q_1, q_2\}$, {0,1}, {Z,0}, δ, $q_0$, Z, $\{q_0\}$), де:

$\delta(q_0, 0, Z) = \{(q_1, 0Z)\}$
$\delta(q_1, 0, 0) = \{(q_1, 00)\}$
$\delta(q_1, 1, 0) = \{(q_2, \varepsilon)\}$
$\delta(q_2, 1, 0) = \{(q_2, \varepsilon)\}$
$\delta(q_2, \varepsilon, Z) = \{(q_0, \varepsilon)\}$

Робота автомата полягає в копіюванні в магазин початкових нулів із вхідного ланцюжка і наступному вилученню по одному нулю з магазина на кожну прочитану одиницю.

### *Регулярні множини і регулярні вирази*

Нехай V- скінченний алфавіт. Рекурсивно ***регулярна множина*** в алфавіті V визначається так:

1) $\varnothing$ – регулярна множина в алфавіті V;
2) $\{e\}$ – регулярна множина в алфавіті V;
3) $\{a\}$, де $a \in V$ – регулярна множина в алфавіті V;
4) Якщо $P, Q$ - регулярні множини в V, то $P \cup Q, PQ, P^*$ – регулярні множини;
5) Ніщо інше не є регулярною множиною в V.

Таким чином, множина в V є регулярною тоді і тільки тоді, коли вона є регулярною внаслідок або 1), або 2), або 3), або її можна одержати з них за допомогою операції об'єднання, конкатенації чи ітерації. Регулярні множини можна визначити через регулярні вирази, праволінійні граматики, детерміновані і недетерміновані скінченні автомати.

***Регулярний вираз*** в алфавіті V визначається рекурсивно:

1) $\varnothing$ - регулярний вираз, що позначає регулярну множину $\varnothing$;
2) e - регулярний вираз , що позначає регулярну множину {e};
3) якщо a∈V, то a - регулярний вираз , що позначає регулярну множину {a};
4) якщо p,q - регулярні вирази, що позначають регулярні множини P,Q, то
   - (p+q) регулярний вираз, що позначає регулярну множину P∪Q;
   - (pq) регулярний вираз, що позначає регулярну множину PQ;
   - (p)* регулярний вираз, що позначає регулярну множину P* ;
5) ніщо інше не є регулярним виразом.

Для спрощення запису і для того, щоб в регулярних виразах писати менше дужок, введемо позначення ( $p^+ = pp^*$ ) та пріоритети операцій : ітерація (*), конкатенація, об'єднання (+).

Операцію об'єднання будемо називати ще й операцією альтернативи і позначати вертикальною рискою (|).

Приклади регулярних виразів:

- 01 позначає множину {01}
- 0* позначає множину {0}*
- (0+1)* позначає множину {0,1}*
- (0+1)*011 позначає множину усіх ланцюжків, що складаються з нулів і одиниць і закінчуються ланцюжком 011
- (a+b)(a+b+0+1)* позначає множину усіх ланцюжків з множини {a,b,0,1}*, що починаються з a або b.

Для кожного регулярного виразу існує регулярна мова, а для довільної регулярної мови існують різні регулярні вирази. Будемо говорити, що ***регулярні вирази рівні***, якщо вони позначають однакові регулярні множини.

Нехай $\alpha, \beta, \gamma$ - регулярні вирази. Тоді справджуються ***тотожності*** над регулярними виразами:

1. $\alpha + \beta = \beta + \alpha$
2. $\alpha + (\beta + \gamma) = (\alpha + \beta) + \gamma$
3. $\alpha(\beta + \gamma) = \alpha\beta + \alpha\gamma$
4. $\alpha e = e\alpha = \alpha$
5. $\alpha^* = \alpha + \alpha^*$
6. $\alpha + \alpha = \alpha$
7. $\varnothing^* = e$
8. $\alpha(\beta\gamma) = (\alpha\beta)\gamma$
9. $(\alpha + \beta)\gamma = \alpha\gamma + \beta\gamma$
10. $\alpha \varnothing = \varnothing \alpha = \varnothing$
11. $(\alpha^*)^* = \alpha^*$
12. $\alpha + \varnothing = \alpha$