**6. tétel**

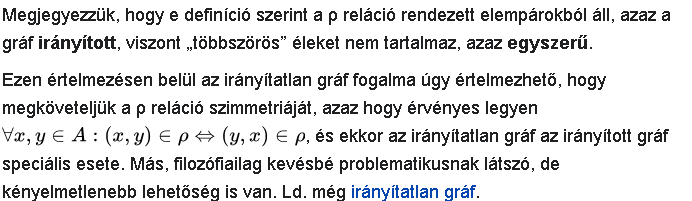
**A gráf fogalma**

A **gráf** a [matematikai](https://hu.wikipedia.org/wiki/Matematika) [gráfelmélet](https://hu.wikipedia.org/wiki/Gr%C3%A1felm%C3%A9let) és a [számítógéptudomány](https://hu.wikipedia.org/wiki/Sz%C3%A1m%C3%ADt%C3%A1studom%C3%A1ny) egyik alapvető fogalma. A gráf csomópontok, csúcsok és rajtuk értelmezett összeköttetések (élek) halmaza. Egy gráfot megadhatunk csúcsainak és éleinek felsorolásával, vagy szemléletesebben egy diagram formájában, ahol a pontok felelnek meg a gráf csúcsainak, az őket összekötő ívek pedig az éleknek. A két megadási mód ekvivalens, azaz a gráf pusztán egy struktúra, semmilyen megjelenítési információt nem tartalmaz, így különböző diagramok is tartozhatnak ugyanahhoz a gráfhoz.

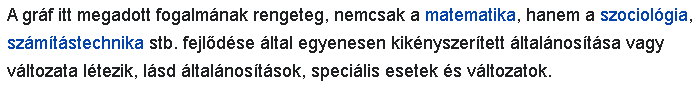
Alapértelmezésben a gráf irányítatlan, azaz nem teszünk különbséget „A-ból B-be”, illetve „B-ből A-ba” menő élek között. Ezzel szemben az irányított gráfokban (angolosan: digráf) a két iránynak irányított élek felelnek meg.

Szintén alapértelmezésben, a gráf csúcsai **címkézettek**, azaz meg lehet különböztetni őket. Bizonyos problémák azonban könnyebben kezelhetők, ha nem különböztetjük meg a csúcspontokat. Persze egy-egy csúcspont így is megkülönböztethető maradhat egyéb jellemzőik alapján, mint például a vele szomszédos csúcsok száma. Hasonlóan, a gráf élei alapértelmezésben címkézettek, de előfordulhat, hogy ezt nem követeljük meg. Az olyan gráfok, amikben sem a csúcspontok, sem az élek nem címkézettek, **címkézetlen** gráfok. Megjegyzés: a „címkézés” szó más kontextusban is elfordul a gráfoknál, itt most az élek-csúcsok megkülönböztetésére szolgáló címkékkel foglalkoztunk.

**Definíció:** Adott egy A [halmaz](https://hu.wikipedia.org/wiki/Halmaz), és egy rajta értelmezett {\displaystyle \rho \subseteq A\times A} bináris (kétváltozós) [reláció](https://hu.wikipedia.org/wiki/Rel%C3%A1ci%C3%B3). Ekkor a {\displaystyle G=\left(A,\rho \right)} párt, vagyis az A halmaz feletti [relációs struktúrát](https://hu.wikipedia.org/w/index.php?title=Rel%C3%A1ci%C3%B3s_strukt%C3%BAra&action=edit&redlink=1) az A halmaz feletti **gráf**nak nevezzük.



**Definíció:** Az *{\displaystyle A}A* halmazt a *{\displaystyle G=\left(A,\rho \right)}G = (A,p)* gráf *tartóhalmaz*ának vagy *csúcshalmaz*ának mondjuk, és (az angol „vertex”=csúcs szó rövidítéseként) *{\displaystyle V\left(G\right)}V(G)*-vel jelöljük.



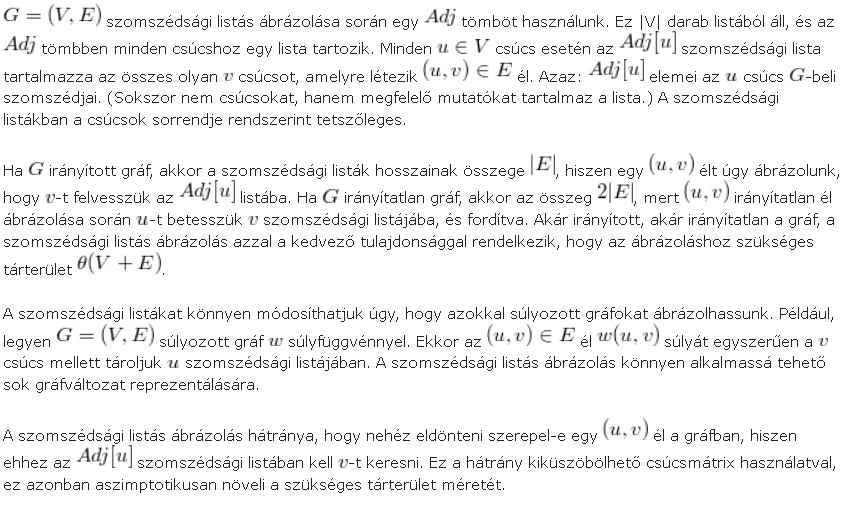
**Alapfogalmak:**

A gráf két élét **szomszédos**nak nevezzük, ha van egy közös csúcspontjuk. Hasonlóan, két csúcspont **szomszédos**, ha van egy közös élük, másként fogalmazva egy éllel vannak összekötve. Egy **séta** szomszédos csúcsok és élek váltakozó sorozata. Az önmagát nem metsző sétát **út**nak hívunk, ha első és utolsó csúcsa különbözik, illetve **kör**nek, ha ez a két csúcs megegyezik. Egy gráf **összefüggő**, ha (élei esetleges irányításáról megfeledkezve) bármely két csúcs között van út.

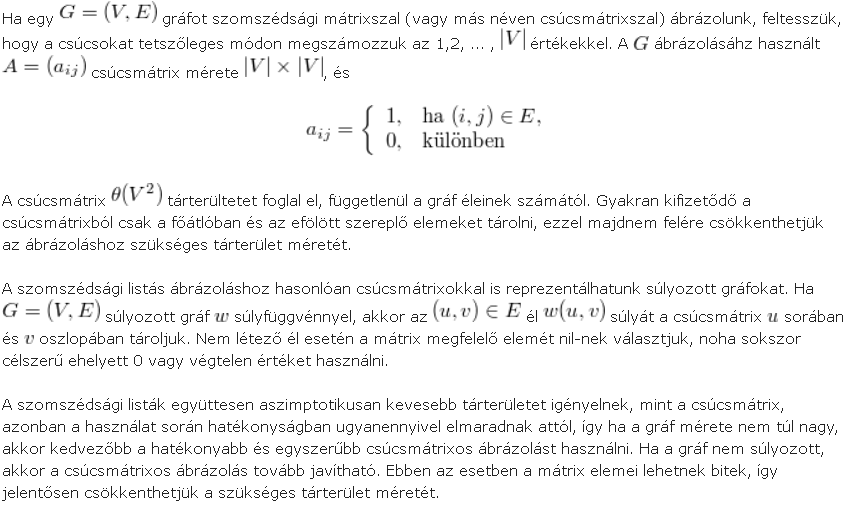
Az ún. **súlyozott** gráfban (ami lehet irányított gráf is), minden élhez hozzárendelünk egy értéket, ami az él költsége, súlya vagy hossza az alkalmazástól függően. Az ilyen gráfok sok helyen előfordulnak, például optimalizálási feladatokban, mint az [utazó ügynök probléma](https://hu.wikipedia.org/wiki/Az_utaz%C3%B3_%C3%BCgyn%C3%B6k_probl%C3%A9m%C3%A1ja).

**Gráf megadásának módjai:**

**Szomszédsági listás:**



**Szomszédsági mátrixos:**



**Nevezetes séták**

*Tudunk-e olyan sétát tenni a gráf élein, hogy*

* *minden élen pontosan egyszer megyünk végig?* (lásd: [Königsbergi hidak problémája](https://hu.wikipedia.org/wiki/K%C3%B6nigsbergi_hidak_probl%C3%A9m%C3%A1ja), [Euler-kör](https://hu.wikipedia.org/wiki/Euler-k%C3%B6r))
* *minden csúcsot pontosan egyszer érintünk?* (lásd: [Hamilton-út](https://hu.wikipedia.org/wiki/Hamilton-%C3%BAt))

Bár a két kérdés hasonlónak tűnik, az első megválaszolására van gyors (lineáris idejű) algoritmus (néha a diagram alapján ránézésre is eldönthető), míg a második az egyik ismert legnehezebb probléma ([NP-teljes](https://hu.wikipedia.org/w/index.php?title=NP-teljes&action=edit&redlink=1)).

[**Legrövidebb út probléma**](https://hu.wikipedia.org/wiki/Legr%C3%B6videbb_%C3%BAt_probl%C3%A9ma)

Súlyozott gráfban:

*Melyik a legrövidebb (legkisebb összsúlyú) út*A*-ból*B*-be?*

Például, ha egy valódi úthálózatban, ahol a csúcsok a csomópontok, az élek az útszakaszok, és *A*-ból *B*-be szeretnénk eljutni, de a különböző lehetséges utak nem egyformán kedvezőek, van rövidebb (gyorsabb, olcsóbb), akkor az élekhez az éleknek megfelelő hosszt (időtartamot, árat) rendelve, a válasz a számunkra legkedvezőbb út.

### Irányítatlan gráf

A *G* irányítatlan gráfot a *G*=(*V*, *E*) rendezett párral jellemezzük, ahol

* *V* a csúcsok halmaza (melyről általában feltesszük, hogy véges) és
* *E* az irányítatlan éleknek megfelelő csúcsok rendezetlen párjainak halmaza.

Az *e*={*u, v*} élről azt mondjuk, hogy *u* és *v* ***között fut***, ***összeköti*** *u*-t és *v*-t.

### Irányított gráf

A {\displaystyle {\overrightarrow {G}}=(V,{\overrightarrow {E}})} irányított gráf:

* {\displaystyle {\overrightarrow {E}}} az irányított élek végpontjai rendezett párjának halmaza

Az *e*=(*u, v*) élről azt mondjuk, hogy *u*-ból indul és *v*-be ***megy***, *v* az *u* ***közvetlen leszármazott***ja (***gyereke***), *u* a *v* ***közvetlen őse*** (***szülője***).

Ha egy irányított gráf nem tartalmaz [irányított kört](https://hu.wikipedia.org/w/index.php?title=Ir%C3%A1ny%C3%ADtott_k%C3%B6r&action=edit&redlink=1), akkor [irányított körmentes gráfnak](https://hu.wikipedia.org/wiki/Ir%C3%A1ny%C3%ADtott_k%C3%B6rmentes_gr%C3%A1f) vagy angol nevének rövidítése szerint DAG-nak (*directed acyclic graph*) hívjuk.

**Egyszerű gráf:**

Az olyan gráfokat amelyekben sem többszörös élek, sem hurokélek nincsenek ***egyszerű gráf***nak hívjuk. Ekkor *E* valódi halmaz és 

**Általánosítások**

* A ***hurokél*** olyan él, amelynek mindkét végpontja megegyezik.
* A gráfokban megengedhetünk ***többszörös-*** vagy ***párhuzamos él***eket, melyek végpontjai megegyeznek. Ehhez az élek halmazát [multihalmazra](https://hu.wikipedia.org/w/index.php?title=Multihalmaz&action=edit&redlink=1), vagy más többszöri előfordulást lehetővé tevő struktúrára kell cserélnünk. Az olyan gráfot, amiben a többszörös élek (és esetleg a hurokélek) megengedettek, **multigráf**nak vagy **pszeudográf**nak hívjuk.
* Néha olyan gráfokat is megengednek, amiben olyan élek is vannak, amiknek csak egy végük van „fél-él”, vagy egy csúcshoz sem kapcsolódnak „szabad él”, például az [előjeles gráfoknál](https://hu.wikipedia.org/w/index.php?title=El%C5%91jeles_gr%C3%A1f&action=edit&redlink=1).
* [Hipergráf](https://hu.wikipedia.org/wiki/Hipergr%C3%A1f) – a hipergráfban egy él kettőnél több csúcspontot is összeköthet.
* [Végtelen gráf](https://hu.wikipedia.org/w/index.php?title=V%C3%A9gtelen_gr%C3%A1f&action=edit&redlink=1)
* Egy irányítatlan gráf felfogható [szimpliciális komplexusnak](https://hu.wikipedia.org/w/index.php?title=Szimplici%C3%A1lis_komplexus&action=edit&redlink=1), ami 1-[szimplexekből](https://hu.wikipedia.org/wiki/Szimplex) (élekből) és 0-szimplexekből (csúcspontokból) áll. Ilyen értelemben ezek a komplexusok a gráfok általánosításai, mert magasabb dimenziójú szimplexeket is tartalmazhatnak.
* Minden gráfból felírható egy [matroid](https://hu.wikipedia.org/wiki/Matroid) (pontosabban: [grafikus matroid](https://hu.wikipedia.org/wiki/Grafikus_matroid)), de általában a gráfot nem lehet visszaállítani a matroidjából, ezért a matroidok nem valódi általánosításai a gráfoknak.

**Teljes gráf:**

A **teljes gráf** olyan [egyszerű gráf](https://hu.wikipedia.org/wiki/Egyszer%C5%B1_gr%C3%A1f), amelynek minden csúcsa össze van kötve minden más csúccsal. Az *n* csúcsú teljes gráfot {\displaystyle K\_{n}}-nel jelöljük [Kazimierz Kuratowski](https://hu.wikipedia.org/wiki/Kazimierz_Kuratowski) lengyel matematikus emlékére.

## **Tulajdonságok[**[**szerkesztés**](https://hu.wikipedia.org/w/index.php?title=Teljes_gr%C3%A1f&action=edit&section=1)**]**

* {\displaystyle K\_{n}} reguláris, minden csúcsának [fokszáma](https://hu.wikipedia.org/wiki/Foksz%C3%A1m_(gr%C3%A1felm%C3%A9let)) {\displaystyle n-1}*n - 1*.
* {\displaystyle K\_{n}} összesen {\displaystyle {\binom {n}{2}}={\frac {n(n-1)}{2}}} élt tartalmaz.
* a [Kuratowski-tétel](https://hu.wikipedia.org/wiki/Kuratowski-t%C3%A9tel) szerint [síkbarajzolható](https://hu.wikipedia.org/wiki/S%C3%ADkbarajzolhat%C3%B3) gráf nem tartalmazhat a {\displaystyle K\_{5}}K5 gráffal [topologikusan izomorf](https://hu.wikipedia.org/wiki/Topologikusan_izomorf) részgráfot.
* {\displaystyle K\_{n}} az (n+1)-[szimplex](https://hu.wikipedia.org/wiki/Szimplex) éleit adja.

**Kör:**

[A gráfelméletben a](https://hu.wikipedia.org/wiki/K%C3%B6r_(gr%C3%A1felm%C3%A9let))**[kör](https://hu.wikipedia.org/wiki/K%C3%B6r_(gr%C3%A1felm%C3%A9let))**[élek olyan egymáshoz csatlakozó sorozata, amelyben az élek és pontok egynél többször nem szerepelhetnek, és a kiindulási pont megegyezik a végponttal.](https://hu.wikipedia.org/wiki/K%C3%B6r_(gr%C3%A1felm%C3%A9let))

Jelölések, fogalmak:

A körben szereplő élek száma a kör hossza. Az *n* hosszú kört *Cn*-nel jelöljük. Speciálisan *C*1 neve [hurokél](https://hu.wikipedia.org/wiki/Hurok%C3%A9l), *C*2 [kétszög](https://hu.wikipedia.org/w/index.php?title=K%C3%A9tsz%C3%B6g&action=edit&redlink=1), *C*3 pedig [háromszög](https://hu.wikipedia.org/wiki/H%C3%A1romsz%C3%B6g).

Egy kört *páros körnek* hívunk, ha a hossza páros, különben *páratlan kör*. Egy gráf akkor és csak akkor [páros gráf](https://hu.wikipedia.org/wiki/P%C3%A1ros_gr%C3%A1f), ha nem tartalmaz páratlan kört.

Egy gráf [girthparaméterét](https://hu.wikipedia.org/wiki/Girthparam%C3%A9ter) kör segítségével definiáljuk: a gráfban található legrövidebb kör hossza.

Euler-kör:

Egy kör [*Hamilton-kör*](https://hu.wikipedia.org/wiki/Hamilton-k%C3%B6r), ha minden csúcsot pontosan egyszer érint. Hamilton-kör létezésére vannak elégséges feltételek (Dirac-feltétel, [Pósa-feltétel](https://hu.wikipedia.org/wiki/P%C3%B3sa-t%C3%A9tel), [Ore-feltétel](https://hu.wikipedia.org/wiki/Ore-t%C3%A9tel), [Chvátal-feltétel](https://hu.wikipedia.org/wiki/Chv%C3%A1tal-t%C3%A9tel)), de, mivel a „Van-e adott gráfban Hamilton-kör?” feladat [NP-teljes](https://hu.wikipedia.org/w/index.php?title=NP-teljes_probl%C3%A9m%C3%A1k&action=edit&redlink=1), pontos feltétel megtalálása nem remélhető.

A Hamilton-körrel duális fogalom az [*Euler-kör*](https://hu.wikipedia.org/wiki/Euler-k%C3%B6r) (avagy Euler-vonal); ami olyan zárt élsorozat, mely minden élet pontosan egyszer tartalmaz. Egy összefüggő gráfban akkor és csak akkor létezik Euler-kör, ha minden csúcsának [fokszáma](https://hu.wikipedia.org/wiki/Foksz%C3%A1m_(gr%C3%A1felm%C3%A9let)) páros. Euler-kör létezésével kapcsolatos a [königsbergi hidak problémája](https://hu.wikipedia.org/wiki/K%C3%B6nigsbergi_hidak_probl%C3%A9m%C3%A1ja).

Körmentes-gráf:

*Körmentesnek* (vagy *aciklikusnak*) nevezzük az olyan gráfot, amely nem tartalmaz kört. Ha egy körmentes gráf összefüggő, akkor [*fa*](https://hu.wikipedia.org/wiki/Fa_(gr%C3%A1felm%C3%A9let)). Mivel minden körmentes gráf fák (esetleg egyelemű) halmaza, a körmentes gráfokat *erdőnek* is nevezzük. Az [irányított körmentes gráfok](https://hu.wikipedia.org/wiki/Ir%C3%A1ny%C3%ADtott_k%C3%B6rmentes_gr%C3%A1f) jelentős szerepet játszanak a [számítástudományban](https://hu.wikipedia.org/wiki/Sz%C3%A1m%C3%ADt%C3%A1studom%C3%A1ny).

**Páros gráf:**

**Páros gráf**nak, **kétrészes gráf**nak vagy **páros körüljárású gráf**nak nevezünk egy *G*{\displaystyle G}iGGg  [gráfot](https://hu.wikipedia.org/wiki/Gr%C3%A1f), ha {\displaystyle G}*G* csúcsainak [halmazát](https://hu.wikipedia.org/wiki/Halmaz) fel tudjuk úgy osztani egy *A*{\displaystyle A}*AAa* és *B*{\displaystyle B} halmazra, hogy az összes *G*{\displaystyle G}*Gg*-beli élre teljesül, hogy az egyik végpontja *A*{\displaystyle A}-ban van, a másik pedig *B*{\displaystyle B}-ben. Egy *G*{\displaystyle G} páros gráfot következőképpen jelölünk: {\displaystyle G}{\displaystyle =}*G = (A, B)*{\displaystyle (A,B)}.

Páros gráf minden részgráfja is páros. Minden [fa](https://hu.wikipedia.org/wiki/Fa_(gr%C3%A1felm%C3%A9let)) páros gráf.

[Teljes páros gráfnak](https://hu.wikipedia.org/wiki/Teljes_p%C3%A1ros_gr%C3%A1f) nevezünk egy olyan páros gráfot melyben minden *A*{\displaystyle A}-beli pont össze van kötve minden *B*{\displaystyle B}-beli ponttal. Jelölés: *Ka,b*{\displaystyle K\_{a,b}}, ahol *a = |A|* {\displaystyle a} {\displaystyle =}{\displaystyle |A|}és *b = |B|*{\displaystyle b}{\displaystyle =}{\displaystyle |B|}.

**Fa, fák, erdő**

A [gráfelméletben](https://hu.wikipedia.org/wiki/Gr%C3%A1felm%C3%A9let) **fának** vagy **fagráfnak** nevezzük azokat a [gráfokat](https://hu.wikipedia.org/wiki/Gr%C3%A1f), amelynek bármely két csúcsát *pontosan egy* út köti össze, azaz a fák körmentes [összefüggő](https://hu.wikipedia.org/wiki/Gr%C3%A1felm%C3%A9let#%C3%96sszef%C3%BCgg%C5%91s%C3%A9g) gráfok. **Erdőnek** nevezzük azokat a gráfokat, amelynek bármely két csúcsát *legfeljebb egy* út köti össze, azaz ahogy az elnevezés is utal rá, az erdő olyan gráf, aminek komponensei fák, vagy ami ezzel ekvivalens, az erdők körmentes gráfok.

* Minden fa [páros gráf](https://hu.wikipedia.org/wiki/P%C3%A1ros_gr%C3%A1f). Minden fa, amelynek [megszámlálható sok](https://hu.wikipedia.org/wiki/Sz%C3%A1moss%C3%A1g) csúcspontja van, [síkgráf](https://hu.wikipedia.org/wiki/S%C3%ADkbarajzolhat%C3%B3_gr%C3%A1f).
* Minden összefüggő G gráfnak van feszítő fája, azaz létezik hozzá olyan fa, ami tartalmazza a G összes csúcspontját, és amelynek élei egyben a G gráfnak is élei. Továbbá minden gráfnak van feszítő erdője, tehát létezik hozzá olyan erdő, aminek a komponensei azonosak az eredeti gráféival.
* Minden fának, amelynek van legalább két csúcspontja, van legalább két levele, azaz van legalább két olyan csúcsa, amelynek a [fokszáma](https://hu.wikipedia.org/wiki/Foksz%C3%A1m_(gr%C3%A1felm%C3%A9let)) 1.
* Egy fa csúcsainak száma 1-gyel nagyobb az élek számánál. Erdő esetén a csúcsok és az élek számának különbsége a komponensek száma.
* Egy erdő {\displaystyle \chi \_{s}(G)} [csillagkromatikus száma](https://hu.wikipedia.org/wiki/Csillagkromatikus_sz%C3%A1m) a fakomponensek közül a legnagyobb sugár+1, illetve 3, amelyik ezek közül a maximális.

**Generatív nyelvtanok**

Egy generatív nyelvtan a jelsorozatok transzformációs szabályait leíró szabályok halmazából áll. A nyelvet alkotó jelsorozatok létrehozásához szükséges, hogy legyen egy egyedi „kezdő” szimbólum, ezután csak a szabályokat kell egymás után alkalmazni (bárhányszor, tetszés szerinti sorrendben) a kezdő szimbólum átalakítására. A nyelv azokból a jelsorozatokból áll, amelyeket az említett módon elő lehet állítani. A szabályok alkalmazásának megengedett lehetőségei közül bármilyen különleges sorrend alkalmazásával az átalakításokkal létrehozhatók jelsorozatok, de ha ezek közül a jelsorozatok közül egyet is többféleképpen is elő lehet állítani, akkor a nyelvtant [kétértelműnek](https://hu.wikipedia.org/w/index.php?title=K%C3%A9t%C3%A9rtelm%C5%B1_nyelvtan&action=edit&redlink=1) nevezik.

Egy *W*=(*V*, *Ax*, *H*) generatív rendszerből úgy kapunk generatív nyelvtant, ha a *V* ábécét felbontjuk közös elemet nem tartalmazó nem üres (és *V* végessége miatt véges) *N* és *T* részhalmazokra ( *V*=*N*∪*T*, *N*∩*T*=∅, *N*≠∅, *T*≠∅), az *Ax* pedig egyetlen, egybetűs, *N*- beli (általában *S*- el jelölt) elemből áll, s minden *H*- beli szabály baloldala legalább egy *N*- beli betűt tartalmaz. Formálisan tehát a generatív grammatika (vagy generatív nyelvan) definíciója:

* + 1. **Definíció.** A *G*=(*N*, *T*, *S*, *H*) rendezett négyest *generatív nyelvtan*nak (vagy *generatív grammatikának*) nevezzük, ha *N* és *T* diszjunkt véges ábécék, *S*∈*N*, *H*⊂(*N*∪*T*)\**N*(*N*∪*T*)\*×(*N*∪*T*)\*. Az *N* elemeit *nemterminális jel*eknek vagy változó szimbólumoknak nevezzük, és általában nagybetűkkel (*S*, *A*, *B*, *C*, …) jelöljük. A *T* elemeit *terminális jel*eknek, vagy konstansoknak nevezzük, és általában kisbetűkkel (*a*, *b*, *c*, …) jelöljük. A *H* elemeit képező (*p*, *q*) rendezett párokat (mint korábban is) *helyettesítési szabály*oknak nevezzük, és általában *p* → *q* alakban írjuk. Az *S* egy kitüntetett nemterminális jel, amely a *G* nyelvtanban a generálás kiinduló vagy kezdő eleme, másnéven *mondatszimbólum*a (startszava). ★

Most definiáljuk, hogy hogyan állítunk elő egy nyelvet egy generatív nyelvtan segítségével, először most is (közvetlen) levezethetőséget kell definiálnunk, ez tulajdonképpen megegyezik a generatív rendszerekben meghatározott azonos fogalmakkal:

* + 1. **Definíció.** Egy *G* generatív nyelvtanban az *r*∈(*N*∪*T*)\* szóból *egy lépésben, vagy közvetlenül levezethető* a *t*∈(*N*∪*T*)\* szó, ha van olyan *p* → *q* helyettesítési szabály a *H*- ban és *p*1, *p*2∈(*N*∪*T*)\* úgy, hogy *r*=*p*1*pp*2 és *t*=*p*1*qp*2. Jelölés: *r*⇒*Gt*, vagy ha egyértelműen meghatározható a *G*, akkor *r*⇒*t*.

Egy *G* generatív nyelvtanban az *r* szóból *levezethető* a *t* szó, ha van olyan *r*0, *r*1, …, *rn* véges szósorozat a (*N*∪*T*)\*- ban, amelyre teljesül, hogy *r*0=*r*, *rn*=*t* és *ri*⇒*ri*+1 (*i*=0, 1, …, *n*-1). Ezt a relációt *r*⇒\**t*- vel jelöljük. Megegyezés szerint minden *r*∈(*N*∪*T*)\* szóra *r*⇒\**r* teljesül.

A *G*=(*N*, *T*, *S*, *H*) *generatív nyelvtan által generált nyelven* a *T*\*- beli szavak következő halmazát értjük: *L*(*G*)={*w*|*S*⇒\**w*, *w*∈*T*\*}. ★

Úgy is mondhattuk volna, hogy *L*(*G*)=*Lg*(*G*)∩*T*\*, ahol *Lg*(*G*) a *G* mint generatív rendszer, pontosabban a (*V*, {*S*}, *H*) generatív rendszer által generált nyelvet jelöli. (Tehát vigyázat: Nem tévesztendő össze a *G* generatív nyelvtan és a *G* mint generatív rendszer által generált nyelv, hiszen *L*(*G*)⊆*Lg*(*G*), vagyis a két nyelv nem esik egybe: *S*∈*Lg*(*G*), de *S*∉*L*(*G*) !) *Lg*(*G*) elemeit, vagyis azon (*N*∪*T*) feletti szavakat amelyek az *S* modatszimbólumból levezethetőek, *mondatformáknak, N*\* elemeit *nemterminális szavaknak, T*\* elemeit pedig *terminális szavaknak,* vagy a nyelvtani analógia miatt *mondatoknak* is hívjuk. Egy adott nyelvtan esetén a levezetés során mondatformák szereplenek, így a levezetett terminális szót sokszor egyszerűen csak levezetett szónak fogjuk hívni (amennyiben ez nem zavaró).

Chomsky nyelvészként a természetes nyelvek leírását szerette volna elérni generatív nyelvtanok segítségével, így a fogalom megfelel annak, hogy a változók a nyelvi fogalmak (ige, főnév, stb.), a konstansok pedig a szótári szavak elemeinek absztrakciói. Habár a természetes nyelvek teljes formális leírása ilyen formán a mai napig nem született meg, a generatív nyelvtanok szerepe igen fontossá vált a mesterséges nyelvek, így a számítógépek fejlődésével, pl. a programnyelvek formális leírásakor.

Első példánkban kezdetleges számítógépes nyelvészeti leírással próbálkozunk.

Tegyük fel például, hogy egy ábécéhez a '{\displaystyle a}*a*' és a '*b*{\displaystyle b}*bb*' szimbólumok tartoznak, a kezdő szimbólum pedig legyen az '*S*{\displaystyle S}*S*' és adottak a következő szabályok:



Kezdő szimbólumunk a „*S*{\displaystyle S}”, de ezután kiválaszthatjuk, hogy melyik szabályt alkalmazzuk a jelsorozat következő elemének előállításához. Ha az 1-es szabályt választjuk, akkor annak alapján az '*S*{\displaystyle S}' szimbólumot a '*aSb*{\displaystyle aSb}'-al helyettesítjük, eredményül tehát a „*aSb*{\displaystyle aSb}*a*” jelsorozatot kapjuk. Ha most ismételten az 1-es szabály alkalmazását választjuk, akkor helyettesítjük az '*S*{\displaystyle S}' szimbólumot a '*aSb*{\displaystyle aSb}*a*'-vel, és akkor a „*aaSbb*{\displaystyle aaSbb}” jelsorozatot kapjuk. Ezt az eljárást addig ismételhetjük, amíg az ábécé szimbólumai megengedik (esetünkben '*a*{\displaystyle a}' és '*b*{\displaystyle b}'). Befejezve a példát, most válasszuk a 2-es szabályt, helyettesítsük az '*S*{\displaystyle S}' szimbólumot a '*ba*{\displaystyle ba}' jelsorozattal, eredményül pedig a „*aababb*{\displaystyle aababb}” jelsorozatot kapjuk, és ezzel be is fejeztük.

Az adott szimbólumokat használva a fentieket sokkal egyszerűbb formában is leírhatjuk:



A nyelvtan által meghatározott nyelv nem lesz más, mint az összes olyan jelsorozat halmaza, amelyeket ezzel az eljárással elő tudunk állítani:



**Chomsky-féle nyelvosztályok**

A nyelvtan és az általa generált nyelv definíciója szerint minden nyelvtanhoz egy egyértelműen

meghatározott nyelv tartozik, de megfordítva, egy nyelvet nem csak egy nyelvtannal generálhatunk.

Két nyelvtant (*gyengén*) *ekvivalens*nek nevezünk, ha ugyanazt a nyelvet generálják, vagy az általuk

generált nyelv legfeljebb az üresszóban tér el. Tehát, formálisan, *G*1 és *G*2 (gyengén) ekvivalens, ha

*L*(*G*1)∖{*λ*}=*L*(*G*2)∖{*λ*}.

Az ekvivalencia fogalmának ismeretében kézenfekvőnek látszik a különböző nyelvtanokat bizonyos

formai tulajdonságok alapján osztályokba sorolni. Az osztályozás alapját a helyettesítési szabályok

alakjára vonatkozó megszorítások képezik abban a hierarchiában, amelyet az elmélet egyik

megalapozója, N. Chomsky vezetett be, és amelyet alább ismertetünk.

**3. Definíció.** A *G*=(*N*, *T*, *S*, *H*)- t *i-típusú nyelvtan*nak (*i*=0,1,2,3) nevezzük, ha az alábbi megszorítások

közül az *i*- edik teljesül:

(0) *Mondatszerkezetű nyelvtan,* (*Phrase-structure*). A generatív nyelvtan korábban már ismertetett

általános definíciójánál feltettük, hgbogy *H* olyan (*N*∪*T*)\*- beli párokból áll, melyek első eleme (azaz a

szabály baloldala) tartalmaz legalább egy nemterminálist. *H*- ra ebben az esetben (azaz *i*=0 esetén)

ezen kívül nem rovunk ki külön feltételt.

A *G*=(*N*, *T*, *S*, *H*) *generatív nyelvtan által generált nyelven* a *T*\*- beli szavak következő halmazát értjük: *L*(*G*)={*w* | *S*⇒\**w*, *w* ∈ *T*\*}. ★

Úgy is mondhattuk volna, hogy *L*(*G*)=*Lg*(*G*)∩*T*\*, ahol *Lg*(*G*) a *G* mint generatív rendszer, pontosabban a (*V*, {*S*}, *H*) generatív rendszer által generált nyelvet jelöli. (Tehát vigyázat: Nem tévesztendő össze a *G* generatív nyelvtan és a *G* mint generatív rendszer által generált nyelv, hiszen *L*(*G*)⊆*Lg*(*G*), vagyis a két nyelv nem esik egybe: *S* ∈ *Lg*(*G*), de *S*∉*L*(*G*) !) *Lg*(*G*) elemeit, vagyis azon (*N*∪*T*) feletti szavakat amelyek az *S* modatszimbólumból levezethetőek, *mondatformáknak, N*\* elemeit *nemterminális szavaknak, T*\* elemeit pedig *terminális szavaknak,* vagy a nyelvtani analógia miatt *mondatoknak* is hívjuk. Egy adott nyelvtan esetén a levezetés során mondatformák szereplenek, így a levezetett terminális szót sokszor egyszerűen csak levezetett szónak fogjuk hívni (amennyiben ez nem zavaró).

Chomsky nyelvészként a természetes nyelvek leírását szerette volna elérni generatív nyelvtanok segítségével, így a fogalom megfelel annak, hogy a változók a nyelvi fogalmak (ige, főnév, stb.), a konstansok pedig a szótári szavak elemeinek absztrakciói. Habár a természetes nyelvek teljes formális leírása ilyen formán a mai napig nem született meg, a generatív nyelvtanok szerepe igen fontossá vált a mesterséges nyelvek, így a számítógépek fejlődésével, pl. a programnyelvek formális leírásakor.

Első példánkban kezdetleges számítógépes nyelvészeti leírással próbálkozunk.

## **Az automata fogalma és főbb típusai**

Automatán egy olyan absztrakt rendszert fogunk érteni, mely egy diszkrétnek képzelt időskála időpillanataiban érkezett ingerek hatására ezen időpillanatokban válasszal reagál, miközben belső állapotát megadott szabályok szerint változtatja a külső ingerek hatására. Az ingerekre adott válasz függ mind az ingerektől mind pedig a pillanatnyi belső állapottól. Ebben az értelemben tehát nemcsak a gépek, hanem bármiféle élő vagy élettelen objektumok tekinthetők automatának, ha ezen séma szerint vizsgáljuk őket, azaz ilyenfajta működést tulajdonítunk nekik.

##### **példa - Automaták**

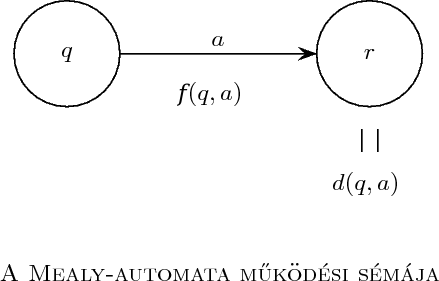
**A legkülönbözőbb létező vagy nem létező dolgok tekinthetők automatának. Automatának tekinthető a lakásunk ajtaja, ablaka. Bizonyos értelemben automatának tekinthető a kedvenc macskánk, a számítógépünk, vagy a lakáscsengőnk is. Vizsgálhatjuk a főnökünket is mint automatát, hisz minden bizonnyal különféleképp reagál arra, hogy az elvárásainak megfelelően cselekszünk-e vagy sem. (És az is valószínű, hogy ezek a dolgok a főnök állapotát is befolyásolják.) De automatának tekinthető az inkák esőistene, aki imádságra esővel, káromkodásra szárazsággal reagál. ★**

**Azon automatákat amelyek egy inputszóhoz egy output szót rendelnek a működésük folyamán, átalakítóknak, transzduszereknek is szokás nevezni. A következőkben főleg ilyenekkel fogunk** **foglalkozni, szemben a későbbi részekben előtérbe kerülő elfogadó automatákkal.**

### Mealy automata

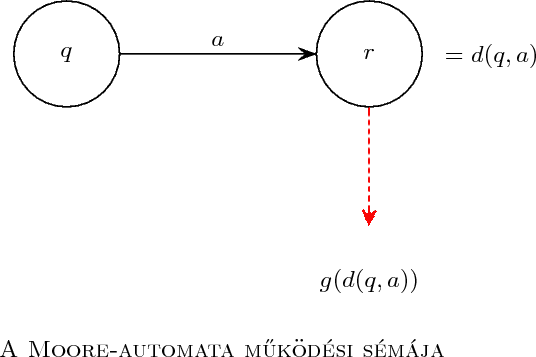
**Az absztrakt automaták egyik nevezetes típusa a Mealy (ejtsd: míli) automata. *Mealy automatán* fogunk érteni egy *A*=(*Q*, *T*, *V*, *d*, *f*) ötöst, aholis *Q* a (belső) állapotok nem üres halmaza, *T* a bemenő jelek nem üres halmaza, *V* a kimenőjelek nem üres halmaza, *d*:*Q*×*T* → *Q* az átmeneti függvény és *f*:*Q*×*T* → *V* a kimeneti függvény.**

**Úgy képzeljük, hogy a Mealy automata diszkrét időskála mentén működik, s annak minden egyes időpillanatában egy-egy jól meghatározott állapotban van. Ha valamely időpillanatban egy *A*=(*Q*, *T*, *V*, *d*, *f*) Mealy automata egy *q*∈*Q* állapotában az *a*∈*T* bemenő jelet kapja, akkor ugyanezen időpillanatban a *f*(*q*, *a*) kimenőjellel reagál, majd a következő időpillanatra átmegy a *d*(*q*, *a*) állapotba.**

****

### Moore automata

**Amennyiben az *A*=(*Q*, *T*, *V*, *d*, *f*) Mealy-automatához létezik olyan *g*:*Q* → *V* függvény, hogy tetszőleges *q*∈*Q* állapota és *a*∈*T* bemenő jele esetén teljesül a *f*(*q*, *a*)=*g*(*d*(*q*, *a*)) egyenlőség, akkor *Moore-automatáról* beszélünk. A Moore-automatát *A*=(*Q*, *T*, *V*, *d*, *g*) alakban szokás megadni, ahol *g*:*Q* → *V* a Moore-automata *jelfüggvénye.* Tetszőleges *q*∈*Q* állapot esetén azt mondjuk, hogy *g*(*q*) a *q*∈*Q állapotjele.***

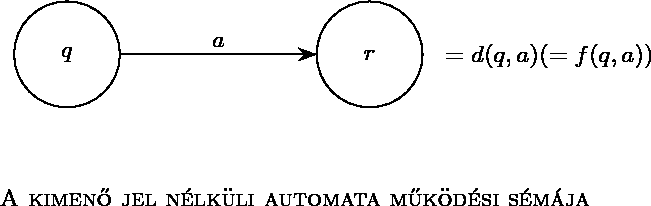
****

**A Moore automata a diszkrét időskála minden egyes időpillanatában egy-egy jól meghatározott *q*∈*Q* állapotban van (amikor az állapotjele *g*(*q*)). Ha egy adott időpillanatban ezen *q*∈*Q* állapotában az *a*∈*T* bemenő jelet kapja, akkor a következő időpillanatban *d*(*q*, *a*) lesz az állapota, s állapotjele pedig *g*(*d*(*q*, *a*)) lesz. Ily módon a Moore-automata egy adott időpillanatban kapott bemenő jel hatására a következő időpillanatra átmegy ezen bemenő jel és a belső állapota által egyértelműen meghatározott állapotba, s ezzel egyidejűleg kiadja az új állapot által egyértelműen meghatározott állapotjelet.**

**Képletesen szólva, amíg a Mealy-automata az olyan embert is modellezheti, aki először cselekszik, azután gondolkodik, addig a Moore-automata az olyan embert modellezi, aki először gondolkodik** **azután cselekszik.**

### Kimenőjel nélküli automata

**A Moore-féle automata a Mealy-féle automata speciális eseteként adódik. A Moore-féle automata további specializálásával jutunk el a kimenőjel nélküli automata fogalmához a következő módon. Amennyiben egy *A*=(*Q*, *T*, *V*, *d*, *g*) Moore-automatára *Q*=*V* és *g*:*Q* → *Q* egy identikus leképezés (azaz minden *q*∈*Q*- ra *g*(*q*)=*q*), akkor a *kimenőjel nélküli automata* fogalmához jutunk. Figyelembe véve azt a tényt, hogy a Moore-automatát egy olyan *A*=(*Q*, *T*, *V*, *d*, *f*) Mealy-automatából származtatjuk, melyhez található olyan *g*:*Q* → *V* függvény, hogy egy tetszőleges *q*∈*Q*, *a*∈*T* pár esetén *f*(*q*, *a*)=*g*(*d*(*q*, *a*)), továbbá figyelembe véve, hogy a kimenőjel nélküli automata olyan Moore-automata, melynek a jelfüggvénye identikus leképezés, azt is mondhatjuk, hogy a kimenőjel nélküli automata egy olyan *A*=(*Q*, *T*, *V*, *d*, *f*) Mealy-automata, melyre *Q*=*V* és *d*=*f*. Ezen okok miatt a kimenőjel nélküli automatát *A*=(*Q*, *T*, *d*) alakban szokás megadni.**

****

**Egy *A*=(*Q*, *T*, *d*) kimenőjel nélküli automata esetén a *T* bemenő jelhalmaz miden *a*∈*T* eleme egy olyan *a*:*Q* → *Q* egy változós műveletnek (vagyis *Q* önmagába történő leképezésének) is tekinthető, mely az állapothalmaz tetszőleges *q*∈*Q* eleméhez az *a*(*q*)=*d*(*q*, *a*) elemét rendeli. Univerzális algebrai kifejezéssel élve tehát a kimenőjel nélküli automaták unoidok, vagy más szóval unáris algebrák. Ez az egyszerű felismerés lehetővé teszi számunkra a modern algebra módszereinek automataelméleti alkalmazását.**

**Amint láttuk, a Moore-féle automata speciális Mealy-automataként definiálható. Később látni fogjuk, hogy ez a specializáció látszólagos, ugyanis információ átalakítás szempontjából a két fogalom ekvivalens. (Az információ átalakításán azt értjük, hogy az automata tetszőleges bemenő információ hatására valamilyen "kimenő" információval reagál.) Nevezetesen, absztrakt szempontból a Mealy és a Moore-féle automaták ekvivalensek egymással abban az értelemben, hogy már a speciálisabb Moore-automatákkal előállíthatók azok az információ átalakítások, amelyek Mealy automatákkal megvalósíthatók. Tehát a Moore-automata ebből a szempontból csak látszólag speciálisabb a Mealy- automatánál.**

**Később látni fogjuk azt is, hogy az elmélet kiépítésénél sok esetben elegendő kimenőjel nélküli automatákra szorítkozni.**

**Az említett három automata típus mindegyike esetén szokás *véges automatáról* beszélni, ha az állapothalmaz, a bemenő jelhalmaz, s a kimenő jelhalmaz végesek. Szokás véges állapotú automatáról vagy *Q*- véges automatáról beszélni, ha az állapothalmaz véges. Hasonló értelemben beszélünk véges bemenetű vagy *T*- véges, illetve véges kimenetű vagy *V*- véges automatáról, valamint (*Q*, *T*)-, (*Q*, *V*)-,** **illetve (*T*, *V*)- véges automatáról.**

### Iniciális automata

**Amennyiben az említett automata-típusok valamelyikénél kijelölünk egy *q*0 iniciális-, vagy más néven kezdőállapotot, s feltételezzük, hogy az automata működésének van egy kezdő időpontja, amikor az automata ebben az állapotban van, akkor iniciális automatáról beszélünk. Az iniciális Mealy- féle automatát *A*=(*Q*, *T*, *V*, *q*0, *d*, *f*) alakban, az iniciális Moore-féle automatát *A*=(*Q*, *T*, *V*, *q*0, *d*, *g*) alakban, illetve az iniciális kimenőjel nélküli automatát *A*=(*Q*, *T*, *q*0, *d*) alakban szokás megadni, ahol mindhárom esetben *q*0 az iniciális állapotot jelöli.**

**Az említett automatáknak szokás beszélni az alábbi általánosításairól is.**

### Parciális és teljesen definiált automata

**Ha az átmeneti, illetve kimeneti függvény lehet parciális is, azaz nem teljesen definiált, akkor *parciális automatáról* van szó. Teljesen definiált függvényértékek esetén időnként szokás *teljesen definiált automatáról* is beszélni.**

##### **példa - Parciális automata**

**Az ajtó szigorú értelemben egy parciális kimenőjel nélküli automatának tekinthető. Bemenő jelei a csukás és a nyitás, s ennek megfelelően két állapota van, nevezetesen csukott és nyitott állapot. Csukott állapotból nyitással lehet nyitott állapotba hozni, nyitott állapotból pedig csukással lehet csukott állapotba hozni. De az ajtó valóban parciális automata, hisz nyitott ajtót kinyitni, vagy csukott ajtót becsukni nem lehetséges. ★**

### Nemdeterminisztikus és determinisztikus automata

**Amennyiben az átmeneti és a kimeneti függvények (illetve Moore-automata esetén az átmeneti és a jelfüggvények) nem egyértelműen definiáltak, *nemdeterminisztikus automatáról* van szó. Ekkor valójában ezek nem is tekinthetőek függvénynek a hagyományos értelemben. Ahhoz, hogy**

**matematikai értelemben mégis függvényekkel dolgozzunk úgy tekintjük őket, mintha nem az állapot-, illetve a kimenő jelhalmazba képeznének, hanem ezen halmazok összes részhalmazainak halmazaiba. Egy nemdeterminisztikus *A*=(*Q*, *T*, *V*, *d*, *f*) Mealy-automata esetén tehát az átmeneti függvény**

***d*:*Q*×*T* → 2*Q*, a kimeneti függvény pedig *f*:*Q*×*T* → 2*V* alakú, ahol 2*Q*, illetve 2*V* az állapothalmaz, illetve a kimenő jelhalmaz részhalmazainak halmazát jelöli. Értelemszerűen, egy nemdeterminisztikus *A*=(*Q*, *T*, *V*, *d*, *g*) Moore-automata esetén az átmeneti függvény *d*:*Q*×*T* → 2*Q*, a jelfüggvény *g*:*Q* → 2*V***

**alakú, míg egy nemdeterminisztikus kimenőjel nélküli *A*=(*Q*, *T*, *d*) automatánál az átmeneti függvény**

**formája *d*:*Q*×*T* → 2*Q*.**

##### **példa - Nemdeterminisztikus automata**

**Nemdeterminisztikus kimenőjel nélküli automatának tekinthető a dobókocka, melynek egyetlen bemenő jele a feldobás. Ezen bemenő jel, azaz a feldobás hatására a dobókocka a hat lehetséges állapotából átmehet a hat lehetséges állapot bármelyikébe annak megfelelően, hogy a feldobás után éppen melyik lapjára esik. ★**

**További változata a nemdeterminisztikus automatáknak, ha megengedjük, hogy az automata bemenő jel nélkül is állapotot váltson: *d*:*Q*×(*T*∪{λ}) → 2*Q*. Ezeket szokás *üresszóátmenetes* (*nemdeterminisztikus*) *automatáknak* is nevezni.**

**A nemdeterminisztikus automata ellentéteként beszélünk determinisztikus automatáról is. Determinisztikus automata esetén tehát a szóban forgó függvényertékek mindig pontosan egy (parciális automata esetén maximum egy) meghatározott értéket vesznek fel. (A nemdeterminisztikus terminológiával pedig a függvények értékkészlete csak egyelemű, illetve maximum egyelemű részhalmazokat tartalmaz.) Determinisztikus automatáknál nem fordulhat elő üresszóátmenet.**

**Ha például egy nemdeterminisztikus *A*=(*Q*, *T*, *V*, *d*, *f*) Mealy-automata esetén a *d*(*q*, *a*) függvényérték a *Q*- nak egy hat elemű részhalmaza, *f*(*q*, *a*) pedig a *V* egy két elemű részhalmaza, akkor az *A* Mealy- automata a *q* állapotából az *a* bemenő jel hatására ezen hat elemű részhalmaz bármelyik elemébe átmehet, s kimenőjelként pedig az említett két elemű halmaz báremlyik elemét kiadhatja. S tekintettel arra, hogy egy halmaznak az üres halmaz is részhalmaza, az is előfordulhat, hogy valamely *q*∈*Q* állapotra és *a*∈*T* bemenő jelre *d*(*q*, *a*), vagy éppen *f*(*q*, *a*) értéke az üres halmaz. Ha *d*(*q*, *a*) az üres halmaz, ez annak felel meg, hogy erre az állapota és bemenő jelre nincs értelmezve egyetlen állapot sem amibe átmenet történhet, ha pedig *f*(*q*, *a*) az üres halmaz, akkor ez azt jelenti, hogy erre az állapota és bemenő jelre nincs értelmezve egyetlen kimenőjel sem.**

**Ezek szerint a parciális automata olyan nemdeterminisztikus automatának tekinthető, ahol a megfelelő függvényértékek vagy egy elemű halmazokat, vagy pedig az üres halmazt szolgáltatják, a teljesen definiált determinisztikus automata pedig egy olyan nemdeterminisztikus automata, ahol ezek a** **függvényértékek mindig egy elemű halmazok.**

### Sztochasztikus automata

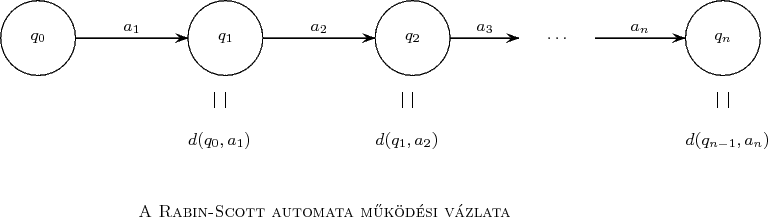
**Fontos általánosítás a *valószínűségi vagy sztochasztikus automata,* mikoris egy *P*((*r*, *b*)|(*q*, *a*)) feltételes valószínűség adja meg, hogy mi a valószínűsége annak, hogy az automata a *r* állapotba megy át és a *b* kimenőjelet adja ki azon feltétel mellett, hogy miközben a *q* állapotban volt, az *a* bemenő jelet kapta. Tehát egy valószínűségi automata *A*=(*Q*, *T*, *V*, *P*) alakban adható meg, ahol *Q* az állapotok nem üres halmata, *T* a bemenő jelek nem üres halmaza, *V* a kimenőjelek nem üres halmaza, *P* pedig** **az említett feltételes valószínűség.**

### Rabin-Scott automata

**Az automaták egy fontos osztályát képezik a *Rabin-Scott féle automaták* (ejtsd rabinszkott), melyeket *felismerő* vagy *elfogadó automatáknak* is hívnak. A Rabin-Scott féle automata egy *A*=(*Q*, *T*, *q*0, *d*, *F*) ötös, ahol *Q* a nem üres állapothalmaz, *q*0∈*Q* a kezdőállapot, *T* a nem üres bemenő jelhalmaz, *d*:*Q*×*T***

**→ *Q* az átmeneti függvény, *F*⊆*Q* pedig a végállapotok nem üres halmaza. ( *q*0∈*F* megengedett, azaz előfordulhat, hogy a kezdőállapot egyúttal végállapot is.)**

**A bemenő jelekből felépülő véges hosszúságú láncokat *bemenő szavaknak* hívjuk. Bemenő szónak tekintjük a *λ* üresszót is, mely nem tartalmaz egyetlen betűt sem. Egy Rabin-Scott automata a *λ* üresszót definíció szerint akkor *ismeri fel,* ha *q*0∈*F*. Egy nem üres, *a*1, …, *an* (nem feltétlenül különböző) bemenő jelekből álló *a*1⋅⋅⋅*an* bemenő szó esetén akkor mondjuk, hogy a tekintett *A*=(*Q*, *T*, *q*0, *d*, *F*) Rabin-Scott féle automata *felismeri,* ha alkalmas *q*1, …, *qn* állapotaira *q*1=*d*(*q*0, *a*1), …, *qn*=*d*(*qn*-1, *an*) teljesülése mellett *qn*∈*F*.**

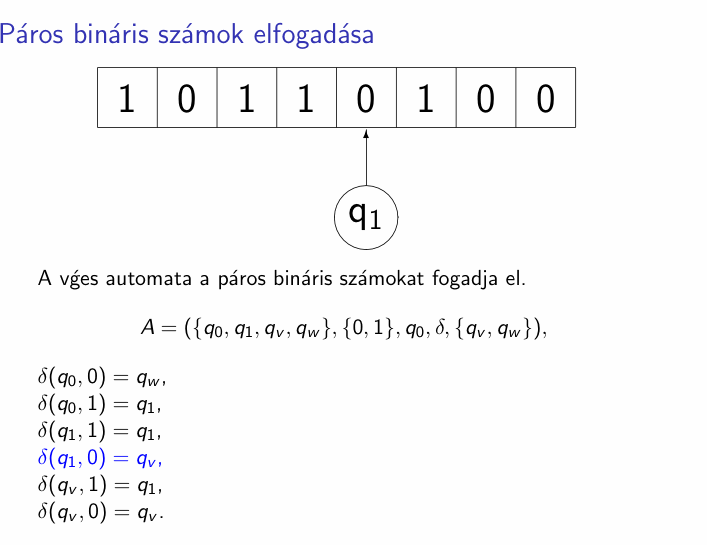
****

**Az *A Rabin-Scott automata által felismert* L*(*A*) nyelvnek* hívjuk mindazon bemenő szavak halmazát, melyeket az automata felismer.**

**Értelmezhetjük a nemdeterminisztikus felismerő automatát is. Ekkor az automata által elfogadott nyelv alatt azon *w* szavak halmazát értjük, amelyekre az automatának van olyan lehetséges állapot- lánca, amely a kezdőallapotból indul és végállapottal végződik, valamint az átmenetek bemenő jeleit összeolvasva éppen a *w* szót kapjuk.**

**Itt jegyezzük meg, hogy az átmenetfüggvényt szokás a *d* helyett a görög δ betűvel is jelölni.**

##### **példa - Rabin-Scott automata működés közben**

****

**★**

**A véges elfogadó automatákra a következő részben, a** [**reguláris nyelvek**](#_bookmark56) **kapcsán még visszatérünk.**

## **Az automaták megadása**

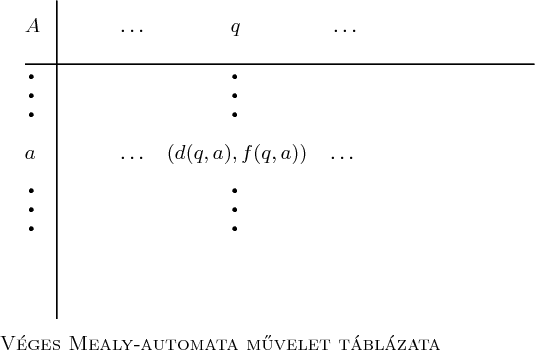
**Egy automatát akkor tekintünk adottnak, ha a hozzá tartozó halmazok és függvények adottak. Egy automatát tehát úgy lehet megadni, hogy megadjuk a hozzá tartozó halmazokat és függvényeket. Ezen halmazok és függvények minden olyan típusú megadása lehetséges, ami a halmazok és függvények** **megadásánál szokásos.**

### Véges automaták megadása Cayley táblázattal

**Véges halmazok és függvények megadásánál szokásos a művelettáblával történő megadás. Automaták művelettáblás megadását automaták Cayley táblázatának (ejtsd: kéjli) is hívjuk Cayley francia matematikus emlékére és tiszteletére, aki véges csoportok művelettábláinak leírására vezette be ezt a táblázatos módszert.**

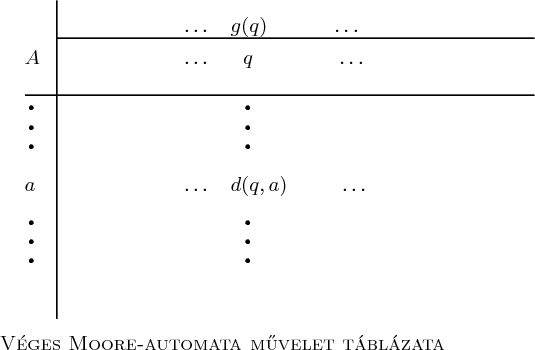
#### Véges Mealy-automata Cayley táblája

**A táblázat bal felső sarkába írjuk az automata nevét, első sorában felsoroljuk az állapotait, első oszlopában pedig a bemenő jeleit. A táblázat (*i*+1)- edik sorában és (*j*+1)- edik oszlopában szerepel egy két dimenziós vektor, melynek első tagja azt mondja meg hogy az automata a *j*- edik állapotból az *i*- edik bemenő jel hatására melyik állapotába megy át, a másik tagja pedig azt mutatja, hogy a *j*- edik állapot az *i*- edik bemenő jel hatására milyen kimenőjelet ad ki.**

****

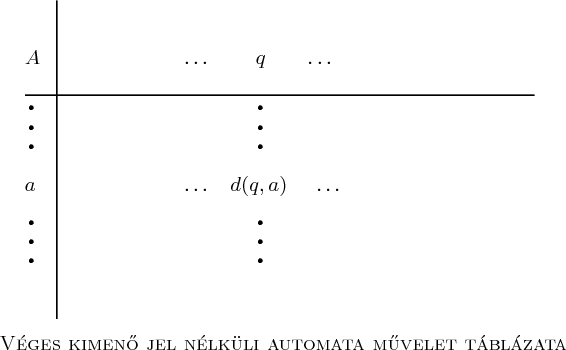
#### Véges Moore-automata Cayley táblája

**A táblázat bal felső sarkába írjuk az automata nevét, első sorában felsoroljuk az állapotait, első oszlopában pedig a bemenő jeleit. Minden állapot fölé beírjuk az állapotjelét. Így az első sor két rész- sorra oszlik. A táblázat (*i*+1)- edik sorában és (*j*+1)- edik oszlopában szerepel az az állapot, amibe az automata a *j*- edik állapotból az *i*- edik bemenő jel hatására átmegy.**

****

#### Véges kimenőjel nélküli automata Cayley táblája

**A táblázat bal felső sarkába írjuk az automata nevét, első sorában felsoroljuk az állapotait, első oszlopában pedig a bemenő jeleit. (Ezesetben az állapotjel maga az állapot, így nem kell az első sorban levő állapotok fölé írni az állapotjelet mint az előző esetben.) Itt is a táblázat (*i*+1)- edik sorában és (*j*+1)- edik oszlopában szerepel az az állapot, amibe az automata a *j*- edik állapotból az *i*- edik bemenő jel hatására átmegy.**

****

##### **4.5. példa - A csengő, mint automata**

**Vegyünk egy villamos csengőt. *a*1 és *a*2 jelöljék azon helyzeteket, mikoris nyomjuk vagy nem nyomjuk a csengőt. A csengő kezdetben az *q*0 kezdőállapotban van, ami annak felel meg, hogy nem cseng. Az *a*1 jel hatására, vagyis a csengő megnyomására átmegy a csengő a *q*0 állapotból a *q*1 állapotba. Megszakítva a csengő nyomását, vagyis az *a*2 jel hatására a csengő átmegy nem csengő, azaz az *q*0 állapotba. Leírásunkat táblázatba foglalva jutunk el a csengő következő absztrakt modelljéhez:**

|  |  |  |
| --- | --- | --- |
| ***A*** | ***q0*** | ***q1*** |
| ***a1*** | ***q*1** | ***q*1** |
| ***a2*** | ***q0*** | ***q0*** |

**A táblázat mutatja, hogy mely jel hatására mely állapotból mely állapotba megy át az automata. Például a 3. sor 3. oszlopában levő *q*0 azt jelenti, hogy *a*2 hatására a *q*1 állapotból az *q*0 állapotba megy át az automata. Táblázatos megadásnál iniciális automata esetén rendszerint az első oszlop jelzi a kezdőállapot oszlopát (azaz annak megadását, hogy különféle bemenő jelek hatására a kezdőállapotból mely állapotokba megy át az automata). ★**

**Megjegyezzük, hogy némely feladatoknál célszerű a táblázatos megadásban a sorok és oszlopok szerepeinek felcserélése, vagyis ekkor az oszlopok a bemenőjeleket (illetve a *λ*- t, ha az**

**automata bemenőjel nélkül is állapotot válthat), míg a sorok az automata állapotait reprezentálják. Mindenképpen célszerű jelezni a táblázat bal felső sarkában, hogy a bemenőjelek, illetve az állapotok hol találhatóak. Elfogadó automaták esetén a végállapotokat is meg kell jelölni, pl. bekeretezéssel. A biztonság kedvéért a kezdőállapotot külön is jelezhetjük, pl. egy nyilacskával.**

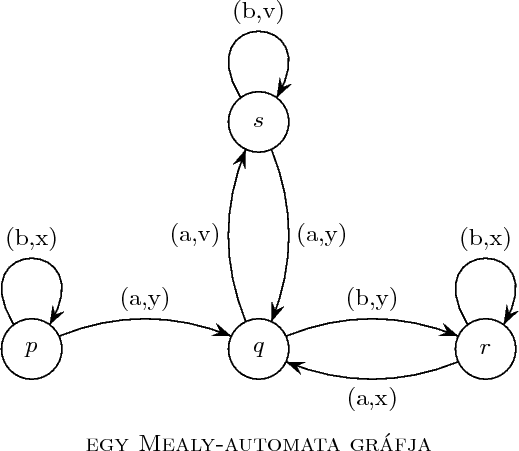
**Parciális automata esetén a táblázat néhány helye üresen maradhat, amit ∅ jelölhet. Itt jegyezzük meg, hogy nem-determinisztikus automaták esetén a táblázat cellái az állapotok-, illetve a kimenőjelek halmazának részhalmazait tatralmazzák.**

### Véges automaták megadása gráfokkal

**Véges automaták megadásának egy másik szokásos módja a címkézett irányított gráffal történő megadás.**

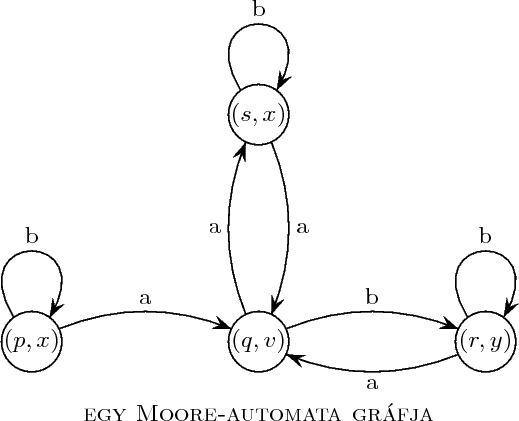
#### Véges Mealy-automata megadása gráffal

**A gráf minden egyes csúcsa meg van címkézve egy állapottal, a csúcsokból kivezető minden irányított él (mely hurokél is lehet) pedig meg van címkézve egy két dimenziós vektorral. Ezen vektor első komponense egy bemenő jel, a második pedig egy kimenőjel. Determinisztikus esetben minden csúcsból pont annyi él vezet ki, ahány bemenő jel van és az élek, valamint azok címkéi adják meg, hogy egy adott állapotból egy adott bemenő jel hatására az automata milyen kimenőjellel reagál és melyik állapotba megy át. Nevezetesen, a bemenőjeleket az élek címkéinek első komponense, a kimenőjeleket az élek címkéinek második komponense, az állapot átmeneteket pedig az élek kezdő- és végcsúcsainak címkéi adják meg. Egy él kezdőcsúcsában lévő állapot az él címke első (bemenő jel) komponense hatására épp abba az állapotba megy át, mellyel az él végcsúcsa van megcímkézve.**

****

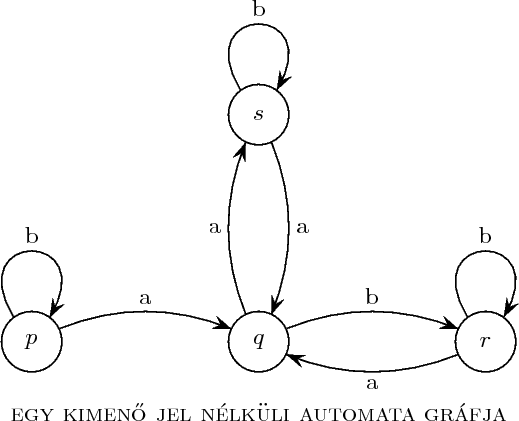
#### Véges Moore-automata megadása gráffal

**A gráf minden egyes csúcsa meg van címkézve egy két dimenziós vektorral, melynek első komponense egy állapot, a második komponense pedig ezen állapot állapotjele. A csúcsokból kivezető minden irányított él (mely hurokél is lehet) meg van címkézve egy bemenő jellel. (Determinisztikus esetben itt is) minden csúcsból pont annyi él vezet ki, ahány bemenő jel van és az élek, valamint azok címkéi adják meg, hogy egy adott állapotból egy adott bemenő jel hatására az automata melyik állapotba megy át. Nevezetesen, a bemenőjeleket az élek címkéi, az állapot átmeneteket és az állapotjeleket pedig az élek kezdő- és végcsúcsainak címkéi adják meg. Egy él kezdő csúcsában lévő címke első (állapot) komponense az él címke (bemenő jel) hatására épp abba az állapotba megy át, ami az él végcsúcsában levő címke első (állapot) komponense. Az átmenet utáni állapotjel pedig az él végcsúcsában levő címke második (kimenőjel) komponense.**

****

#### Véges kimenőjel nélküli automata megadása gráffal

**Csaknem olyan szerkezetű gráffal történik a megadás mint a Moore-automata esetén. Az egyedüli lényeges különbség, hogy a csúcsok itt az állapotokkal vannak megcímkézve (és nem két dimenziós vektorokkal mint a Moore-automata esetén). Tehát a gráf minden egyes csúcsa meg van címkézve egy állapottal, a csúcsokból kivezető minden irányított él (mely hurokél is lehet) pedig meg van címkézve egy bemenő jellel. Most is igaz a (teljesen definiált) determinisztikus esetre, hogy minden csúcsból pont annyi él vezet ki, ahány bemenő jel van és az élek, valamint azok címkéi adják meg, hogy egy adott állapotból egy adott bemenő jel hatására az automata melyik állapotba megy át. Nevezetesen, a bemenőjeleket az élek címkéi, az állapot átmeneteket pedig az élek kezdő- és végcsúcsainak címkéi adják meg. Egy él kezdőcsúcsában lévő állapot címke az él címke (bemenő jel) hatására épp abba az állapotba megy át, ami az él végcsúcsában levő állapot címke értéke.**

****

**Itt jegyezzük meg, hogy elfogadó automata esetén szokás a végállapotokat pl. dupla körrel rajzolni, míg a kezdőállapot jelölésére szokás az adott állapotba egy plusz bemenő nyilat rajzolni.**

## **Veremautomaták**

Ha a véges automata definíciójában az állapothalmaz végességére vonatkozó követelményt elhagyjuk, akkor (a véges ábécéjű) végtelen automata fogalmához jutunk. Ez a fogalom e formájában túl általánosnak bizonyult, ezért bevezették a végtelen automaták speciális fajtáit.

##### példa - Verem működése

Az egyetemi menzán az ételeket tálcán szokás a kiszolgáló pultoktól az asztalokhoz vinni. A tálcák a kiszolgáló pultoknál egymás tetején vannak elhelyezve, s mindig csak a legfelső tálcát lehet elvenni. Ha a legfelső tálca valamiért nem szimpatikus (színe vagy más miatt), ahhoz hogy az alatta levő valamelyik tálcához hozzáférjünk, le kell venni a legfelsőt, s félretenni. Ezt a mozdulatot mindaddig folytatnunk kell, míg a kívánt (valamiért szimpatikus) tálcához nem jutunk. Ezután (ha rendeseknek akarunk látszani) akkor a félretett tálcákat visszapakoljuk (többnyire fordított sorrendben, mint ahogy elvettük) úgy, hogy minden visszapakolt tálca fölé helyezzük a következő visszapakolandót. ★

A veremautomatát eredetileg aritmetikai kifejezések számítógéppel történő kiértékelésére vezették be, s történeti érdekessége, hogy egy veremautomatát realizáló szoftver volt az első olyan számítógépes szoftver termék, mely szabadalmi oltalmat kapott az USA-ban. Egyben ez volt a világon az első szoftver szabadalom. (A szabadalmaztatás a hatvanas években történt.)

A verem olyan alapvető adatszerkezet, amely fontos szerepet játszik a programozás során is. A számítógép, anélkül, hogy erre konkrét veremkezelő utasításokat használnánk alkalmazza a verem adatszerkezetet a rekurzív programozási módszerek esetén. Tekintsük a következő C-szerű pszeudo- kódot, amely a jól ismert Hanoi-tornyai probléma megoldására írodott.

##### példa - Verem, mint rekurziós eszköz

Adott *n* páronként különböző méretű korong és három rúd. Kezdetben mind az *n* korong az első (source) rúdon van. Bármelyik rúdon bármelyik pillanatban a korongok csak méretüknek megfelelő sorrendben lehetnek: adott korong alatt nála kisebb sohasem lehet. A cél hogy az összes *n* korongot átrakjuk az első rúdról a második (target) rúdra a harmadik (help) rúd alkalmas segítségével, oly módon, hogy minden lépésben egy korongot (a legfelsőt) tudjuk áthelyezni egy rúdról egy másik tetejére az előbb ismertetett feltételt betartva. A megoldás során egyes rudakhoz, mint verem adatszerkezethez tudunk hozzáférni.

function Hanoi (n,s,t,h){ if (n>0){

Hanoi (n-1,s,h,t); movedisk (s,t); Hanoi (n-1,h,t,s);}

}

Magyarázat: a Hanoi(*n,s,t,h*) hívásával *n* korongot helyezünk át az *s* rúdról a *t* rúdra úgy, hogy közben a *h* rudat segédrúdként használhatjuk.

Másrészt a megoldás során a rekurzív hívások ugynacsak egy verem segítségével hajtódnak végre, mindig a megfelelő helyre visszadva a vezérlést. Ahogy az előző kód kompaktságán is látszik a rekurzió segítségével nagyon tömör forráskóddal tudunk feladatokat megoldani. ★

A veremautomata esetén egy potenciálisan végtelen befogadóképességű veremmemória összes lehetséges tartalma eredményez végtelen sok állapotot. A veremmemóriát úgy képzelhetjük el, mint egy pozíciókra felosztott, egyirányban végtelen szalagot.

Minden pozícióba egy-egy jel írható. A jelek kiolvasása, illetve törlése a bevitelhez képest fordított sorrendben történik (LIFO: Last In First Out adatszerkezet). Más szóval, a verem belsejének tartalmához közvetlenül nem férünk hozzá, hanem mindig csak a verem tetején lévő jelet tudjuk kiolvasni, illetve módosítani (felülírni, vagy törölni) és csak a verem tetejére (a benn lévők fölé) helyezhetünk el újabb jelet.

A verem alján kezdetben csak egy speciális szimbólum van, a kezdő veremszimbólum.

Általában az input szalaggal ellentétben, amit vízszintesen elhelyezkedőnek gondolunk, a vermet legtöbbször függőleges elrendezésűnek képzeljük/rajzoljuk. A formális leírásunkban a hellyel való takarékosság miatt, vízszintesen úgy fogjuk elképzelni, hogy a legelső betű a legfelső; vagyis mintha ez a szalag balra volna végtelen.

A veremautomata a véges bemenő szót egy input szalagon kapja meg, melyről egy olvasófejjel, balról jobbra haladva betűnként tudja leolvasni a bemenő szót. Az is megengedett, hogy az input szalag üres legyen, azaz az üresszót tartalmazza.

A veremautomatához a potenciálisan végtelen veremmemórián és az input szalagon kívül tartozik még egy véges iniciális nemdeterminisztikus kimenő jel nélküli automata, mely az input szalagról beolvasott betű, a verem tetején levő betű, valamint a belső állapota alapján fogja a verem tetejét és belső állapotát megváltoztatni. Az is megengedett, hogy egy-egy ilyen elemi lépés során az input szalagon a soron következő betűt ne vegye figyelembe, és/vagy az átmenet lényegében ne függjön a verem tetején lévő betűtől. Azt az elemi lépést, mikor a veremautomata az input szalagon soron következő betűt nem olvassa be (az olvasó fej helyben marad), *λ*- lépésnek is szokás hívni. Ezt a véges iniciális nemdeterminisztikus kimenő jel nélküli automatát a fejezet további részében a rövidség kedvéért a vereamutomata véges automatájának (vagy véges vezérlőjének) hívjuk.

A veremautomata működésének kezdetekor a verem szinte üres (csak a kezdő veremszimbólum van a verem alján), az input szalag olvasófeje a szalag első betűjére mutat (vagy ha az input szalag tartalma az üresszó, ezt érzékeli), s a veremautomatához tartozó véges automata pedig a kezdő állapotában van.

A működés veremautomata esetén is diszkrét időskála mentén haladva történik. Ha a veremautomatához tartozó véges automata a működés során a teljes input elolvasásával eljut egy végállapotba, akkor veremautomata megáll. A veremautomata megáll akkor is, ha a hozzá tartozó véges automata egy olyan állapotban van, melyhez nem tartozik egyetlen alkalmas átmenet sem. Az alkalmas átmenet létezése azt jelenti, hogy a verem tetején lévő jel figyelembe vételével vagy anélkül és az input szalagon lévő következő betű (ha van olyan) figyelembe vételével vagy anélkül az adott állapotból van lehetséges átmenet egy másik állapotba a veremautomata véges automatájában. A veremautomatához tartozó véges automata belső állapotát a továbbiakban röviden a veremautomata belső állapotának, vagy a veremautomata állapotának mondjuk. (Szigorúan véve a veremautomata belső állapotát egy pár határozza meg, melynek első eleme a verem tartalma, a második pedig a veremautomatához tartozó véges automata belső állapota, s így tekintve a veremautomata végtelen.)

Ha az input szalagon egy nem üresszó van, s a veremautomata úgy áll meg végállapotban, hogy előzőleg az input szalag utolsó betűjét is beolvasta, akkor azt mondjuk, hogy a veremautomata az input szalagon levő szót elfogadta. Ha a veremautomata úgy áll meg végállapotban, hogy az input szalag tartalma az üresszó, akkor azt mondjuk, hogy a veremautomata az üresszót elfogadta. Akkor mondjuk, hogy a (nemdeterminisztikus működésű) veremautomata egy bemenő szót elfogad, ha

van olyan futása (működésmódja), hogy a bemenő szót elfogadja. A veremautomata által elfogadott bemenő szavak összeségét a veremautomata által elfogadott nyelvnek hívjuk. A veremautomatáknak többféle definíciója is ismert, a következőkben többféle (általánosabb és egyszerűbb modellt is megvizsgálunk). Látni fogjuk, hogy a veremautomaták által elfogadott nyelvek osztálya épp a környezetfüggetlen nyelvek osztálya.

Formálisan, egy *veremautomata* (PushDown Automaton) a következő hetes: *PDA*=(*Q*, *T*, *Z*, *q*0, *Z*0, *d*, *F*) ahol

*Q* a belső *állapotok* nem üres és véges *halmaza,* vagy más néven állapot ábécé,

*T* az *inputábécé, vagy szalagábécé*, *Z* a *veremábécé,*

*q*0∈*Q* a *veremautomata kezdő állapota, Z*0∈*Z* a *verem kezdőjele* (*"alja"*)*,*

*d* leképezés a *Q*×(*T*∪{*λ*})×*Z*- ból a *Q*×*Z*\* véges részhalmazaiba a (nemdeterminisztikus) *átmeneti függvény,*

*F*⊆*Q* a *veremautomata végállapotainak halmaza.*

Egy veremautomata pillanatnyi konfigurációján értjük azt a (*u*, *q*, *z*) hármast, ahol *u*∈*T*\* az input még fel nem dolgozott része, *z*∈*Z*\* a veremmemória pillanatnyi tartalma ( *z* első betűje a verem tetején lévő betű), *q*∈*Q* pedig a veremautomata pillanatnyi belső állapota. Egy veremautomata minden lépésben a pillanatnyi konfigurációból a *d* átmeneti függvény definíciója értelmében egy vagy több pillanatnyi konfigurációba mehet át (vagy épp nem mehet át egybe sem). Mint már említettük, az átmenet történhet oly módon, hogy közben a veremautomata nem kéri a bemenő jelsorozat következő jelét ( *λ*- lépés).

A *PDA* veremautomata egy (*au*, *q*, *Xz*)∈(*T*\*×*Q*×*Z*+) konfigurációt *egy lépésben átalakít* a (*u*, *p*, *tz*)∈(*T*\*×*Q*×*Z*\*) konfigurációba, ahol *a*∈(*T*∪{*λ*}), *X*∈*Z* pedig a verem tetején levő szimbólum, (jelekben: (*au*, *q*, *Xz*)⊦*PDA*(*u*, *p*, *tz*), vagy csak egyszerűen: (*au*, *q*, *Xz*)⊦(*u*, *p*, *tz*)), ha van olyan *a*∈*T*∪{λ}, *p*, *q*∈*Q*, valamint *t*∈*Z*\*, hogy a következő összefüggés fennáll: (*p*, *t*)∈*d*(*q*, *a*, *X*).

Egy veremautomata a (*u*, *q*, *z*)∈(*T*\*×*Q*×*Z*+) konfigurációt (véges lépésben) *átalakít* (átalakíthat) a (*v*, *p*, *t*) konfigurációba - jelekben: (*u*, *q*, *z*)⊦\*(*v*, *p*, *t*) - ha van olyan véges konfigurációsorozat *P*0, *P*1, …, *Pn*, melyenk első tagja *P*0=(*u*, *q*, *z*), utolsó tagja *Pn*=(*v*, *p*, *t*), és bármely két egymást követő tagjára *Pi*⊦*PDAPi*+1 (*i*=0,…,*n*-1) fennáll.

A veremautomata által (végállapottal) elfogadott nyelv:

*L*(*PDAf*)={*w*∈*T*\*|(*w*, *q*0, *Z*0)⊦\*(*λ*, *p*, *z*) valamely *z*∈*Z*\* és *p*∈*F* esetén }.

Látható, hogy a veremautomata egy lépésben egy jelpáron tud operációt végrehajtani, aholis a jelpár egyik tagja a veremábécé egy jele, a második tagja pedig az input ábécé egy betűje, vagy az üresszó.

Az eddigiek során a veremautomata végállapottal fogadta el a nyelvet, a szakirodalomban ugyanilyen jól ismert az üres veremmel elfogadó automata. Sok esetben, mint pl. a rekurzív programok esetén is, akkor tekintjük a rendszer működését helyesen befejezettnek, ha végül kiürül a verem.

Egy *PDA*=(*Q*, *T*, *Z*, *q*0, *Z*0, *d*, *F*) veremautomata üres veremmel fogadja el az

*L*(*PDAe*)={*w*∈*T*\*|(*w*, *q*0, *Z*0)⊦\*(*λ*, *p*, *λ*) valamely *p*∈*Q* esetén } nyelvet.

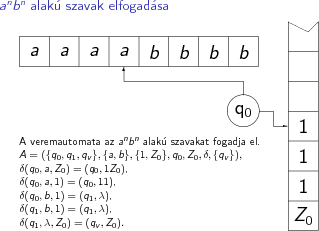
Egy üres veremmel elfogadó veremautomata esetén (vagyis ha az *L*(*PDAe*) nyelvre vagyunk kíváncsiak), tulajdonképpen az *F* végállapotok halmazára nincs is szükség.

A veremautomatákat is megadhatjuk grafikusan a véges automatákhoz hasonló módon: körökkel jelöljük az állapotokat, külön (egy bemenő nyíllal) megjelölve a kezdőállapotot. Végállapottal elfogadó automata esetén duplakarikákkal jelöljük a végállapotokat, üres veremmel elfogadó automata esetén pedig nincs végállapot (vagyis a végállapotok hiányával jelezzük, hogy üresveremmel elfogadásról van szó). A *d* átmenetfüggvény alapján, ha (*p*, *t*)∈*d*(*q*, *a*, *X*), akkor egy irányított él vezet a *q* állapotból a *p* állapotba és az élen az (*a*, *X*/*t*) címke ( *a*∈*T*∪{*λ*}, *X*∈*Z*, *t*∈*Z*\* ) mutatja, hogy az átmenet akkor lehetséges ha *a*- t olvasunk az input szalagról miközben *X* van a verem tetején, amit az átmenet során a *t*- vel helyettesítünk.

Egy tetszőleges veremautomata esetén a végállapottal, illetve az üres veremmel elfogadott nyelvek nagyon különbözhetnek egymástól. Ezért nagyon fontos, hogy, ha adva van egy veremautomata, akkor legyen adott az is, hogy mely módon akarjuk azt nyelv elfogadásra használni.

##### példa - Végállapottal elfogadó veremautomata működése

Olyan veremautomata működés közben, mely az *anbn* alakú szavakat fogadja el:



★

##### példa - Az azonos számú *a* és *b* betűből álló szavakat végállapottal elfogadó veremautomata

A következő veremautomata olyan szavakat fogad el, melyekben az *a*- k és *b*- k száma megegyezik.

