„KI ITT BELÉPSZ HAGYJ FEL MINDEN REMÉNNYEL”

1. **Definíciók: ábécék, szavak, formális nyelvek, műveletek.**

   Indokolja meg, hogy miért van az, hogy amikor egy determinisztikus automatából reguláris grammatikát készítünk, akkor az algoritmus alapján készült grammatika tényleg pont azokat a szavakat generálja, amelyeket az automata felismer

ábécék: Jelek véges nemüres halmazát nevezzük ábécének. Az ábécét alkotó jeleket az ábécé betűinek nevezzük. Felhasználjuk őket például szavak generálására.

szavak: Egy tetszőleges ábécé betűiből alkotott véges hosszú sorozatokat hívjuk szavaknak.

Az a szó, ami nem tartalmaz betűt az az üres szó jele: ε.

formális nyelvek: Azt a halmazt, amiben egy (tetszőleges) ábécé által alkotott (tetszőleges) szavak vannak formális nyelveknek hívjuk. Tehát L halmaz pontosan akkor formális nyelv, ha létezik egy olyan V_L ábécé, melyre igaz, hogy L részhalmaza V^*_L. És természetesen csak akkor formális nyelv ha van hozzá nyelvtan ami generálja.

Műveletek:

szavak szorzása: ha α szót szorzom β szóval akkor az α szó végére ragasztom a β szót. Ebből következik, hogy két szó szorzása általában nem kommutatív (pl. 1 * 0 = 10 de 0 * 1 = 01, és ez a kettő nem egyenlő) az új szavunk hossza d(α) + d(β) lesz.

szavak hatványozása: ha α szót hatványozom belátható, hogy $\alpha^0 = \varepsilon$, $\alpha^n = \alpha^{n-1}\alpha$ $(n \geq 1)$ (pl. $1^0 = \varepsilon$, $1^3 = 1^2 1 = 111$)

Minden halmaz műveletet használhatunk a nyelvekre:
$L_1 \cup L_2$: az L_1 vagy az L_2 szavai
$L_1 \cap L_2$: az L_1 és L_2 által felismert szavak tartománya
$L_1 \setminus L_2$: azon L_1 szavak melyek nincsenek bent az L_2-ben

Iterálás: Iterálásnak nevezzük egy L nyelv hatványainak egyesítését, jelölése: L^* Igazából az iterálás az L-béli szavak tetszőleges véges sok tényezőiből álló szorzataiból áll (lényegében bármilyen L-ben levő szavak sorrendje egymáshoz ragasztva + az üres szó)

Determinisztikus automatából generált grammatika (4.8-as tétel):

Van egy determinisztikus automatánk, ami

$$A = <K, V, \delta, q_0, F>$$

K az állapotok halmaza,
V a szalag abc,
δ az átmenet függvény,
q_0 a kezdő állapot,
F pedig a végállapotok halmaza.

Ehhez definiálhatunk egy reguláris grammatikát, ami

$$G = <V, K, q_0, P>$$

A Grammatika V-je az automata szalag abc-je ezzel korlátozzuk a grammatikánkat, hogy csak annak az ábécének a betűit használhatja, amiket az automatánk képes felismerni.
A nyelvtanunk K-ja azokat a nem terminális jeleket tartalmazza amilyen állapotok szerepeltek az automatában. Erre azért van szükség mert így nem fordulhat elő, hogy olyan függvénye legyen az automatának, amit a grammatikánk ne tudna generálni mert nincs olyan nem terminális jele, amiből indulhatna.
q_0 a kezdő szimbólum ahogy az automatának is ez a kezdő állapota.
Az általunk definiált grammatika P-je tartalmazza az összes $\delta(q, a) = p$ függvényt.
$q \rightarrow ap$ formában, amennyiben a p eleme a végállapot halmazának akkor $q \rightarrow a$ szabály is a P-be kerül.
Fontos kitérni az üres szóra hiszen, ha q_0 a végállapotokba tartozik előfordulhatna, hogy a grammatikában olyan szabály is előfordul, ahol $q_0 \rightarrow \varepsilon$ és olyan szabályok is, ahol a q_0 a szabály jobb oldalán áll. Azonban a Chomsky féle szabály szerint ez nem lehetséges az általunk tanult reguláris nyelvtanra sem. Ahhoz, hogy ezt elkerüljük be kell vezetni egy q_0' kezdő szimbólumot, amihez az összes q_0 szabályt feljegyezzük és még a $q_0' \rightarrow \varepsilon$ szabályt is. Az így kapott grammatika reguláris és csak azokat a szavakat képes generálni, amiket az 'A' automatánk be tud olvasni.

2. Generatív grammatikák, típusok, a Chomsky féle hierarchia. Indokolja meg, hogy a Chomsky hierarchia miért úgy épül fel, ahogy! Indokolja meg, hogy miért van szükség a Chomsky hierarchiában az „epszilonos” extra kitételre! Indokolja meg, hogy az „epszilonos” extra kitétel szűkíti-e a generálható nyelvek osztályát!

Egy grammatika úgy épül fel alakra, hogy
G=<V,W,S,P>
V – az ábécé, amit a nyelvtanunk használhat ezekből a karakterekből épülhetnek fel a szavak, más néven terminális jelek
W – A nem terminális jelek ábécéje, ezek segédeszközként szolgálnak a generálásnak, ezek a jelek nem lehetnek a generált szóban.
S – A kezdő (betű) szimbólum. Az első nem terminális jel, amivel a szavak felépítése indulhat. Az automata kezdő pontja, ami be tudja olvasni a nyelvtanunkat.
P – Úgynevezett helyettesítési szabályok, a nyelvtanunk szabályai. Egy szabály felépítése milyen szabályokon alapul attól függ melyik nyelvtan csoportba tartozik a Chomsky féle hierarchián belül.

Azt mondhatjuk, hogy G nyelvtan generál egy formális nyelvet, aminek a szavai azok a szavak, amiket a G nyelvtan létre tud hozni. Akkor tudja generálni a kiválasztott szót a grammatika, ha a szó eleme a V*-nak és a kezdő szimbólumból levezethető.

Chomsky féle hierarchia:
A hierarchia négy fajta nyelvtan csoportból tevődik össze

0. Általános
1. Környezet függő
2. Környezet független
3. Reguláris

3 eleme 2 eleme 1 eleme 0-nak

- Általános nyelvtan a legnagyobb 'halmaza' a hierarchiának ennek a szintnek az összes többi szint részhalmaza. Ez a legtágabb értelemben értelmezett nyelvek osztálya (igazából nyelvtanok, mint a magyar vagy ilyenek I guess)

- Környezet függő nyelvek részei az általános nyelvtanoknak azonban valamennyivel limitáltabbak mint az általános nyelvtanok. A szabályai alatt állhat terminális és nem terminális jel is és a jobb oldalán is állhat terminális és nem terminális jel is.
  aAb → aBb, ahol a, b elemei a terminális jeleknek és A eleme a nem terminális jeleknek míg B eleme a nem terminális és terminális jeleknek (Megengedhető az S → ε akkor, ha az S nem áll a szabályok jobb oldalán)

- Környezet független nyelvtanokat és reguláris nyelvtanokat használtunk a legtöbbet a félév alatt. a Környezet független nyelvtanok úgy épülnek fel, hogy a szabályok bal oldalán egy nem terminális jel ál a jobb oldalán pedig nem terminális és terminális jelek tetszőleges sorrendje is állhat
  A → B (Megengedhető az S→ ε akkor, ha az S nem áll a szabályok jobb oldalán)
  Ahol A eleme a nem terminális jeleknek és B eleme a nem terminális és terminális jelek uniojának

- A reguláris nyelvtanok formára úgy néznek ki, hogy a szabályok bal oldalán áll egy nem terminális jel. A szabályok jobb oldalán vagy egy terminális jel áll, vagy egy terminális jel és utána egy nem terminális jel. Így észrevehető, hogy a reguláris nyelvtanok szavaihoz 'jobbról' ragasztjuk a többi 'betűt'.
  A → a vagy A → aA (Megengedhető az S→ ε akkor, ha az S nem áll a szabályok jobb oldalán)
  Ahol A eleme a nem terminális jeleknek és 'a' eleme a terminális jeleknek.

Mielőtt bebizonyítanám, hogy az epszilonos extra kitétel nem szűkíti a generálható nyelvek osztályát tisztába kell lenni egy Lemma-val. A Lemma úgy szól, hogy ha van egy nyelvtan, amiben a kezdő szimbólum a szabályok jobb oldalán is áll akkor képezhető egy vele ekvivalens nyelvtan, ami nem tartalmazza a kezdő szimbólumot a jobb oldalán a szabályoknak.
Ezt úgy tesszük, hogy bevezetünk egy új nem terminális jelet, amit pl S'-nek nevezünk el. Az S' szimbólum örökli az összes szabályát az eredeti nyelvtanunk kezdő szimbólumának. Így az eredeti kezdő szimbólum már nem lesz kezdő szimbólum, és az új szimbólumunk nem állhat a szabályok jobb oldalán, hiszen amíg nem vezettük be nem létezett az eredeti nyelvtanban így szabályai sincsenek. A szabályai azok a szabályok, amit egy másik nem terminális jeltől örökölt. A két nyelvtan továbbra is ugyanazokat a szavakat képes generálni tehát ekvivalenseknek tekinthetőek ebből a szempontból.

Az üresszó problémát ennek a lemmának a segítségével tudjuk megoldani. Ha van egy nyelvtanunk, aminek a kezdő szimbóluma a szabályok jobb oldalán áll akkor nem létezik benne egy olyan szabály, ami az üresszót generálja. Ez lesz a G nyelvtanunk, amiben nincsen benne az üresszó. Ehhez a G-hez létezik egy olyan G' nyelvtan, amit az előbb említett Lemma-val leképezve olyan nyelvtanná tesszük, amiben nincsen a kezdő szimbólum a szabályok jobb oldalán. Ebben az esetben az új kezdő szimbólumhoz tehetünk egy olyan szabályt, ami $S' \rightarrow \varepsilon$. Beláthatjuk, hogy az a szó, ami tartalmazza az $S' \rightarrow \varepsilon$ szabályt az csak az üresszót generálhatja, ami nem mutat ki a nyelvtanunk osztályából mert az üresszó generálásán kívül más szó képzésére ezt a szabályt nem használhatjuk. A két nyelvtanunk így ugyanazokat a szavakat tudja generálni, és ebből következik, hogy mivel mindkét nyelvtannak a szabályai azonosak ezért mindkét nyelvtan ugyanabba a hierarchiában elfoglalt halmazban tud formális nyelvet generálni. Példának okáért, hogy ha a G nyelvtanunk egy reguláris nyelvtan akkor a G' nyelvtanunk is egy reguláris nyelvtan lesz mert sem a Lemma sem az új $S' \rightarrow \varepsilon$ szabály nem változtat a típusán. Így mindkét nyelv reguláris. De ugyanez a bizonyítás alkalmazható a környezet független nyelvtanokra is.

3. **Reguláris nyelvek és tulajdonságaik (zártsági tételek; az automaták elméleténél tanult reguláris nyelvekre vonatkozó tételek is).**
- Indokolja meg, hogy a reguláris nyelvek szorzatára vonatkozó tételnél a konstrukció miért helyes!

Reguláris nyelvek tulajdonságai

Két reguláris nyelv uniozható
Két reguláris nyelv szorozható
Két reguláris nyelv kivonható egymásból
Két reguláris nyelvnek van közös halmaza

$L'' \cup L' = L$

Tegyük fel, hogy van két reguláris nyelvünk, ami L’’(G’’) és L’(G’) mindkettő reguláris nyelvet reguláris nyelvtanok generálják. Ilyenkor képezhető egy harmadik reguláris nyelvtan, ami mindkét formális nyelvet tudja generálni. A séma így szól van a G’’=<V’’,W’’,S’’,P’’> és a G’=<V’,W’,S’,P’>. Készíthetünk egy $G =< V' \cup V'', W'' \cup W' \cup \{S\}, S, P >$. Az S-ből, ami a kezdő szimbólum következni fognak azok a szabályok, amik a S’ és S’’-ből következnek kivéve az S’ → ε és vagy S’’ → ε abban az esetben, ha ilyen szabályok voltak akkor elhagyjuk őket és az S szimbólumhoz egy S → ε szabályt írunk. Ennek az a lényege, hogy ha esetleg lenne szabály, amiben az S’ vagy S’’ a szabályok jobb oldalán állnának úgy, hogy közben generálják az üres szót is akkor nem lehetne helye a Chomsky féle hierarchiában a reguláris nyelvtanok terén. Azonban a tanult lemma alapján ez a plusz szabály nem tesz különbséget a nyelvtan típusában, és a Lemmával tudjuk azt, hogy ez a kezdő szimbólum nem tud a szabályok jobb oldalán állni.
Beláthatjuk, hogy ezzel az új nyelvtanunkkal, ha egy szót választunk, ami eleme az L’’ nyelvnek akkor biztosan lesz a G nyelvtanunknak olyan konstrukciója, ami generálja. Mivel a választott szó eleme az L’’ nyelvnek az L(G) nyelv pedig az L’’ U L’ így biztosan benne van az L(G) nyelvben is az általunk választott szó is. És mivel mindkét nyelv, aminek a szavait az L(G) generálni tudja reguláris, így az L(G) is reguláris, de ez triviális.

L’’L’ = L
Két reguláris nyelv összeszorozható, ha L’’ reguláris nyelvet generálja egy
G’’=<V’’,W’’,S’’,P’’>
és L’ nyelvet generálja egy
G’=<V’,W’,S’,P’>. A G’’ nyelvtan összes olyan szabályához, ami terminális jelhez vezetne, vagyis befejezné a szó generálását hozzá kötjük az L’ nyelvet generáló reguláris nyelvtan kezdő szimbólumát. Ezeket a két nyelvtanhoz készítünk egy új nyelvtant, ami úgy épül fel, hogy
G=<V’UV’’,W’’UW’,S’’,P> A P tartalmazza az összes szabályát a G’ nyelvtannak és az összes szabályát a G’’ nyelvtannak, amiket a fent említett módon módosítottunk.
Azért az S’’ a kezdő szimbólum mert így biztosan az G’’ nyelvtannak a generálási rendszerében kezdjük a szó generálását, és amikor végeznénk a generálással tehát olyan szabályba lennénk, ahol a nem terminális jel generál egy terminális jelet tehát A→a, mindazonáltal az előbb megtett változtatástól ezek a szabályok így néznek ki A→aS’. Ebből következik, hogy az S’-ből csak olyan szavakat tudunk képezni amilyen szavakat a G’ nyelvtan generálni tud.

A véges determinisztikus vagy nemdeterminisztikus automaták felismerik a reguláris nyelvtanokat mert. egy G nyelvtanhoz tudunk készíteni egy olyan A automatát ami így épül fel

$G=<V,W,S,F>$ és $A=<K,V, \delta, q_0, F>$ K tartalmazza az összes állapotot ami igazából annyi amennyi a nem terminális jelek plusz egy E nevű végállapot. Erre azért van szükség mert így minden nem terminális jelnek saját állapota van amiből a saját leképzése tud indulni, így nem fordulhat elő az hogy olyan szabályt kelljen ábrázolni az automatánkon amit nem tudunk mert nincs hozzá állapotunk. V a szalag ábécé, azaz a betűk, amikből a szavak képezve vannak, és a betűk, amiket az automata felismer, egyáltalán be tud olvasni. A nyelvtannak a szalag ábécéjét és az automata szalag ábécéjét is ugyanazzal a betűvel jelöljük azért, mert a kettő ugyanaz a halmaz. A δ tartalmazza az összes olyan szabályt, amit az F tartalmaz a nyelvtanoknál, csak itt az állapotok közötti ugráláshoz használjuk őket. Az automatákat ábrázolva ezek lesznek a nyilacskák, amik összekötik az állapotokat. A q_0 a kezdő állapot, ami ugyanazt a célt szolgálja, amit az S kezdő szimbólum. Ennél a pontnál kezdődik az első betű beolvasása. F pedig a végállapotok halmaza ebben az esetben egy vagy maximum két végállapotról beszélünk. Minden olyan F-ben definiált szabály, aminek a jobb oldalán csak terminális jel van az E végállapotunkba tart, ami felismeri a szót. Még egy végállapotunk lehet és az a kezdő állapot. Ez csak akkor lehetséges, ha a nyelvtan generálja az üresszót. Amennyiben a nyelvtan generálja az üresszót az automatánk kezdőállapota végállapot lesz, de ilyenkor a már említett lemma segítségével meg kell azt oldanunk, hogy a kezdő állapot ne állhasson a szabályok jobb oldalán.

4. Determinisztikus véges automaták komponensei és működése.
- Indokolja meg, hogy miért igaz a Bar-Hillel lemma! Indokolja meg, hogy miért igazak a Bar-Hillel lemma általunk tanult következményei!

Véges determinisztikus automaták komponensei:

A=<K, V, δ, q_0, F> ahol

K = állapothalmaz, a belső állapotok halmaza

V = A bemenő jelek véges halmaza, asszociálható a nyelvtanban levő generatív ábécével, ugyanis az automata is csak ennek az ábécé halmaznak az elemeit tudja befogadni

δ = Szabályok, amik alapján az automatában levő állapotok között ugrálás van, úgy kell elképzelni, mint egy függvényt, amihez a beadott betű hatására rámutat egy állapotra vagy végállapotra, amihez a szó tovább lesz küldve.

q_0 = Kezdő állapot itt kezdődik a beolvasása a betűknek olyan, mint a kezdő szimbólum a grammatikáknál. Innen kezdődik a betűk beolvasása.

F = A végállapotok halmaza, az itt szereplő állapotok azok az állapotok, amihez, ha elér a betűk beolvasása akkor az automata felismeri a beadott szót.

A Bar-Hillel lemmának sok felhasználása van, a kurzus alatt ezt a Lemmát arra használjuk fel, hogy megállapítsuk, hogy egy formális nyelvhez létezik-e egy automata, ami felismeri. Másként megfogalmazva algoritmizálható-e a beolvasása a formális nyelvnek, amit vizsgálunk.
Tegyük fel, hogy van egy formális nyelvünk, amit A-val jelölünk, amihez van egy α szimbólummal jelölt szó, ami eleme ennek a formális nyelvnek. A Bar-Hillel Lemma szerint kell lennie egy olyan küszöbszámnak, amit n_0-val jelölünk, amire igaz, hogy $n_0 \leq \alpha$, ha ez beteljesül akkor az α szót szét tudjuk tagolni három részre ezt a három részt u ,v ,w szimbólumokkal jelöljük. Ilyenkor igaznak kell lennie, hogy $d(uv) \leq n_0$. Miután ezt mind megcsináltuk igaznak kell lennie, hogy a $v \neq \varepsilon$. Még kikötés, hogy minden i számra, ami eleme a pozitív egész számoknak és a nullának hatványelemként használható a v-hez. Tehát $u*v^i*w$ eleme a formális nyelvünknek.

Amennyiben az α szóhoz található egy olyan küszöbszám amit támpontnak nézve fel tudjuk darabolni három részre a szót úgy, hogy a szó első két tagja ne legyen hosszabb a küszöbszámnál amit kijelöltünk. Fontos megjegyezni, hogy a második szelete a szavunknak nem lehet az üresszó. Mert ha üresszó lenne akkor hiába hatványoznánk ezt a szeletet nem számítana, nem bizonyosodna be a feltételünk. Ami fontos a Lemmába, hogy ha van a v mondjuk a hatodik hatványon akkor úgy néz ki a szavunk, hogy
u*v*v*v*v*v*v*w
és ezt a szót is fel kell tudnia ismernie az automatának, hogy beláthassuk, hogy a formális nyelvünk automata által felismerhető.

Példa a Bar-Hillel Lemmára: van egy formális nyelvünk, ami így épül fel
F=<α : $\{a^n b^n\}$, ahol n eleme a pozitív egész számoknak és a nullának>
Ez automata által nem felismerhető. Indirekt bizonyítást fogunk használni hozzá. Abban az esetben, ha az ’a’ betűk ismétlődésének hossza az n_0 küszöbszám akkor a ’b’ betűk hossza is ennek a küszöbszámnak a hossza. Akkor tagoljuk fel három részre ezt a szót. Például legyen hat ’a’ karakter az u-ban és három ’a’ betű a v-ben. w-ben pedig lesz a ’b’ karakterekből kilenc darab. Beláthatjuk hogy ha itt ebben az esetben a v részt hatványozzuk és semmi más nem változik akkor nem teljesül az amit a formális nyelvünk kikötött. Ezen bizonyítás alatt ezt a formális nyelvet nem lehet automatával felismertetni.

5. **Nemdeterminisztikus véges automaták. A determinisztikus véges automatákkal és a reguláris nyelvekkel való kapcsolatuk.**
- Indokolja meg, hogy miért van az, hogy amikor egy reguláris grammatikából nem-determinisztikus automatát készítünk, akkor az algoritmus alapján készült automata tényleg pont azokat a szavakat ismeri fel, amelyek generálhatók a grammatikával!

Véges determinisztikus automaták: Azok az automaták, amik úgy épülnek fel, hogy
$A=<K, V', \delta, q_0, F>$
K az állapotok halmaza, V az ábécé, amit képes beolvasni, δ a szabályok, amik szerint az automata lépked az állapotok között, q_0 a kezdő állapot, F pedig a végállapotok halmaza, ami részhalmaza az összes állapotok halmazának, azaz a K-nak. δ(q, a) hatására egy darab állapotba juthat csak q az egy állapot a pedig a betű, aminek a hatására δ léptet az automata állapotai között.
Véges nemdeterminisztikus automaták: Azok az automaták, amik úgy épülnek fel, hogy
$A=<K, V', \delta, q_0, F>$
K az állapotok halmaza, V' az ábécé, amit képes beolvasni, δ a szabályok, amik szerint az automata lépked az állapotok között, q_0 a kezdő állapot, F pedig a végállapotok halmaza, ami részhalmaza az összes állapotok halmazának, azaz a K-nak. δ(q, a) hatására azonban több állapotba is juthat az automata (q az egy állapot a pedig a betű, aminek a hatására δ léptet az automata állapotai között.) Fontos megjegyezni, hogy ettől függetlenül mindkét fajta automata csak reguláris nyelvtanok által generált nyelveket tud beolvasni.

A véges determinisztikus automaták egy nyelvtan alapján pont azokat a szavakat tudja beolvasni, amiket a nyelvtan generál. Ezt onnan tudhatjuk, hogy ha van egy nyelvtanunk, ami úgy épül fel, hogy G=<V, W, S, F> Ahol V az ábécé amelyik betűket a nyelvtan felhasználja ahhoz, hogy szavakat generáljon. W azok a nem terminális jelek, amik segítik a szavak generálását, de amíg van benne ilyen nem terminális jel addig nem is generálja le a szót. S a kezdő szimbólum amelyik szimbólumból a konstrukció megkezdődik. F pedig a szabályok halmaza, amiket a nem terminális jelek hatására generál a nyelvtan. Sok hasonlóságot lehet felfedezni az automaták és a nyelvtanok felépítése között, nem véletlen ez hiszen minden reguláris nyelvtanhoz elkészíthető egy véges nem determinisztikus vagy véges determinisztikus automata. Az automatánk felépítése $A=<K, V', \delta, q_0, F>$. Ebben láthatjuk, hogy a q_0 ugyanolyan szerepet foglal, mint az S, amikor a nyelvtanunk elkezd generálni az S-ből kezdi a generálást. Amikor az automata elkezdi beolvasni a szót akkor a q_0-t használja arra, hogy léptessen. és annyi 'nyíl' lesz az automatában ahány szabály van a nyelvtanunkban. Tudhatjuk, hogy úgy fogjuk a végállapotokat megkapni, hogy létrehozunk egy végállapotot, amibe az összes olyan nyílt vezetjük, aminek a bal oldalán nem terminális a jobb oldalán meg csak terminális jel lehet. Ez alapján levezetve
$A=<K, V', \delta, q_0, F>$
$K = W \cup \{E\}$
$V' = V$
δ = (q, a) ahol q = W és a = V az összes szabályt tartalmazza tehát a δ halmaz tartalmazza az F halmazt
$q_0 = S$
$F = E$

Fontos még tudni, hogy minden nem determinisztikus véges automatából készíthető determinisztikus véges automata. Ezt úgy érjük el, hogy az állapotokra úgy tekintünk, mint halmazokra. Ha van egy olyan betű aminek a hatására a nem determinisztikus automata több irányba is ugrana az állapotok között akkor van létre lehet hozni egy halmazt amibe a betű hatására ugrik például ha a q_0 pontból a betű hatására az automata a q_0-ba és a q_1-be is ugorhat. Akkor létrehozhatunk egy olyan állapotot, amiben a $\{q_0, q_1\}$ állapot. Ezt a szabályt követve az összes ugrást megnézzük amennyiben egy állapot valamelyik betűre nincs értelmezve. azaz nem mutat nyíl belőle, ami a betű hatására lesz használva egy {üres} állapotot hozunk létre. Ebbe az állapotba fogjuk vezetni az összes olyan nyilat, ami eredetileg az automatánkban nem lenne értelmezve. Ha van egy$\{q_0, q_1\}$ állapot akkor mind a q_0 és a q_1 hatására ugyanannak a szónak a hatására hova ugrana az állapot, és ezt a metodikát követve kell minden ábécé hatására minden állapotot ami a kezdő állapotból elindul leírni. Amennyiben az automata felismeri az üresszót a kezdő állapotot végállapottá kell tenni és mivel a reguláris nyelvtanoknál a kezdő állapot nem állhat a szabályok jobb oldalán akkor, ha a kezdő állapot az üresszóba mutat nem kell foglalkoznunk akkor, hogy mi van akkor, ha önmagába mutatna ilyenkor az automata.

6. Véges automaták minimalizációja.

A minimalizációt csak akkor tudjuk megkezdeni, ha az automatánk teljesen definiált és véges determinisztikus automata. Ilyenkor azokat az állapotokat, amiket a kezdő állapotból nem tudunk elérni nyugodtan elhagyhatjuk.

A minimalizáció két állapot relációján alapul. Van egy automatánk, ami:
$A = <K, V, \delta, q_0, F>$
K – az állapotok halmaza, (azok az állapotok, amik nem elérhetőek a kezdő állapotból azok elhagyhatók)
V – a szalag ábécé amelyik karaktereket az automata be tudja fogadni
δ – az állapotok közötti mozgást biztosító ’szabályok’
q_0 – A kezdő állapot, ahol a betűk beolvasása kezdődik
F – A végállapotok halmaza, ami a K-nak részhalmaza

A két állapot közötti relációkra teljesül az, hogy ekvivalencia relációk. Azaz szimmetrikusak, tranzitívak és reflexívek. Két állapot akkor van egymással relációban, ha egy szó hatására ugyanoda jutnak el.
Reflexív – Minden α szóra igaz, hogy önmagával relációban áll hiszen saját maga ugyanoda jut a szó hatására
Szimmetrikus – Hiszen, ha α szó relációban van β-val akkor igaz az is, hogy β relációban van α val, hiszen a sorrend mindegy mert csak akkor vannak relációban, ha egy szó hatására ugyanabba a pontba érkeznek.
Tranzitív – Hiszen beláthatjuk, hogy ha α szó kap egy γ szót és β is ugyanazt a γ részt kapja toldásnak akkor ugyanabba az állapotba fognak érkezni. Ezt azért tudjuk mert ha α és β relációban vannak akkor mielőtt megkapnák a γ részt már ugyanabban az állapotban vannak, ez a gamma rész pedig saját magával reflexív hiszen ugyanazt a hatást éri el α-val és β-val is.

Ez alapján a gondolatmenet alapján fogjuk az automatánkat minimalizálni.
csinálunk egy K X K mátrixot, aminek minden sora és minden oszlopa egy állapotot jelöl. A mátrixunk fő átlóját és az általunk kiválasztott alsó vagy felső háromszöget figyelmen kívül hagyjuk. A megmaradt háromszögben az összes minden állapot párt megnézünk reláció szempontjából. Amennyiben az egyik állapot végállapot és a másik állapot nem vég állapot a mátrixban a közös blokkba *-ot írunk. Csillaggal jelöljük azokat az automata párokat, amiket sehogy sem fogunk tudni összevonni.
Ez logikus hiszen azokat az állapotok lehetnek potenciálisan összevonva, amelyeknek az automatában a funkciójuk megegyezik.
A többi pár, amihez nem került csillag meg kell vizsgálnunk őket, amennyiben a páros valamelyik betű hatására csillagos párhoz mutat a pár, amit vizsgálunk is csillag lesz. Vannak esetek amikor egy olyan párhoz fog mutatni a betű, amivel vizsgáljuk a párunkat, amit még nem vizsgáltunk. Ilyenkor egy várólistára tesszük a párunkat, amit újra vizsgálunk amint a pár, amihez mutat meg lett vizsgálva. Abban az esetben, ha egy olyan párhoz mutat, aminek mindkettő tagja ugyanaz a tag akkor az figyelmen kívül hagyandó. Miután elvégeztük az összes összehasonlítást így az üres blokkos párokat összevonhatjuk. Amikor az új minimalizált automatánkat összeállítjuk leírjuk az összes állapotot úgy, hogy amik nem üresek tehát nem lettek összevonva azokat leírjuk ahogy eredetileg voltak. Amik pedig össze lettek vonva egy állapotként ábrázoljuk. Ezután az összes szabályt bejelöljük. Ellenőrzésként használhatjuk azt a gondolatmenetet, hogy ha jól vontuk össze az állapotokat akkor mindkét állapot ugyanannak a betűnek a hatására ugyanoda menne tehát nem lesz olyan hogy az automata véges nem determinisztikus lesz.