به نام خدا

جزوه درسی و آموزشی

نظریه زبان ها و ماشین ها

این جزوه برداشتی آزاد (شخصی) از مطالب ارئه شده در کلاس درس نظریه کارشناسی می باشد.

نظریه محاسبات از سه بخش تشکیل شده است:

1- تئوری آتاماتا یا همان نظریه زبان ها و ماشین ها

Atumata theory

2- نظریه محاسبه پذیری

Computability theory

3- نظریه پیچیدگی

Complenity theory

چرا در رشته کامپیوتر که یک رشته کاربردی است نظریه زبان ها و ماشین ها بررسی می شود؟

1) مفاهیم ، نظریه ها و اصولی را بیان می کنند که در درک این رشته به ما کمک می کند ، رشته کامپیوتر حاوی مفاهیم وسیعی از طراحی ماشین تا برنامه سازی است و دارای جزئیات بسیار زیادی است و تنوع زیادی دارد، اما علی رغم تنوع زیاد مفاهیم و اصول مشترکی نیز وجود دارد و برای مطالعه این اصول مدل های انتزاعی از کامپیوتر و محاسبه ساخته شده است ، این مدل ها خواص مشترک سخت افزارها و نرم افزارها و ساختارهای پیچیده ای را که در کار با کامپیوتر با آن بر خورد می کنیم دارا هستند. (مدل سازی سطح بالا)

2) ایده هایی که در این مباحث مطرح می شود مانند نخی اکثر مفاهیم کامپیوتر را به هم پیوند می دهند و به عنوان نمونه طراحی دیجیتال زبان های برنامه سازی و کامپایلرها را می توان نام برد.

3) مسائلی که در این مبحث مطرح می شوند مانند ریاضی باعث باز شدن فکر می شوند. در این مبحث به مدل هایی می‌پردازیم که صفات و مشخصات اصلی کامپیوترها و کاربردهایشان را به نمایش می گذارد. برای مدل کردن سخت افزارها آتاماتا معرفی می شوند.آتاماتا ساختاری است که تمام ویژگی های که یک کامپیوتر را دارا می باشد ، آتاماتا دارای ورودی است ، خروجی تولید می کند ممکن است دارای حافظه موقت باشد و در تولید خروجی از ورودی می‌تواند تصمیم گیری نماید.

زبان صوری

تجریدی از مشخصه های کلی زبان های برنامه سازی است که یک زبان صوری متشکل از مجموعه ای از علایم و قوانین شکل دهی است که توسط آن ها علائم ترکیب شده و به صورت موجودیت هایی به نام جمله در می آید، یک زبان صوری مجموع تمام رشته هایی است که قوانین شکل دهی تائید می کند. زبان هایی که در این جا بحث می شود مدل سطح زبان های برنامه سازی هستند و با مطالعه آن ها می توان بسیاری از مسائل مر بوط به زبان های برنامه سازی را فرا گرفت.

همان طور که گفته شد تئوری آتاماتا تعاریف و خصوصیات مدل های ریاضی محاسبات را ارائه می کند این تئوری دارای رول مدل آتاماتای متناهی fimit atumata و مدل گرامرهای مستقل از متن (context free gramer) است که کاربردهای فراوانی دارد.

تئوری آتاماتا نقطه شروع مباحث نظریه محاسبات است ، مباحث تئوری محاسبه پذیری و تئوری پیچیدگی هر دو بر اساس تئوری آتاماتا بنا شده اند.

در نظریه محاسبه پذیری computability مدلی به نام ماشین تورینگ turing معرفی می شود که ادعا می شود هر عملی که توسط کامپیوتر انجام پذیر است توسط این ماشین هم انجام پذیر است (تز تورینگ) همچنین در این نظریه مسائل قابل حل و غیر قابل حل هم شناخته می شوند و بسیاری از مفاهیم اولیه مرتبط با تئوری پیچدگی نیز مطرح می شوند.

در نظریه پیچیدگی مرتبه هر الگوریتم بررسی می شود و نتیجه گیری می شود الگوریتم هایی فقیرترند که مرتبه پایین تر و در نتیجه زمان و حجم محاسبه کمتری برای حل مسئله داشته باشند.

اجتماع عضو تکراری

$S1 , s2$

$S1 \cup s2 = \{ x : x \in s1 \text{ or } x \in s2 \}$

$S1 \cap s2 = \{ x : x \in s1 \text{ and } x \in s2 \}$

$S1 - s2 = \{ x : x \in s1 \text{ and } x \in s2 \}$

$\bar{s} = \{ x : x \in U \text{ and } x \notin s \}$

مجموعه ای که هیچ عضوی ندارد:

$\emptyset = \{\}$

$S \cup \emptyset = s - \emptyset = s$

$S \cap \emptyset = \emptyset$

$\emptyset = U$

$\bar{S} = s$

- **مجموعه محدود یا متنهای** (finit): مجموعه ای که دارای تعداد عناصر محدود باشند را گویند.
- **مجوعه نا محدود یا نا متنهای** (in finit) : به مجموعه ای که تعداد عناصر آن نا محدود باشند.

برای مجموعه های نا متنهای اگر خاصیت مشترک مشخص و آشکار باشد نیازی به نوشتن رابطه نیست.

$B = \{0, 2, 4, 6, 8\}$

$B = \{ i \mid i = 2k , k \in N \} \cup \{ 0 \}$

زیر مجموعه :

مجموعه s1 زیر مجموعه S نامیده می شود اگر تمامی عناصر s1 در S باشند.

$S1 \subseteq s$

Dis joint در مجموعه مجزا

اگر s1 و s2 دارای هیچ عنصر مشترکی نباشند:

$S1 \cap s2 = \emptyset$

آن ها را دو مجموعه مجزا (dis joint) می نامند.

ü تعداد عناصر مجموعه محدود S به صورت | S | نمایش داده می شود.

یک مجموعه تعداد زیادی زیر مجموعه دارد مجموعه ای که شامل همه زیر مجموعه های مجموعه S باشند. زیر مجموعه توانی S (power set) نامیده می شود و با 2^S نشان داده می شود.

A = { a ,b ,c }

2^A = { {a} , {b} , {c} , {a,b} , {a,c} , {b,c} , {a,b,c} {} }

مجموعه توانی اعضای S

$|s| = n \longrightarrow |2^s| = s^{|s|}$

روابط مربوط در مجموعه ها:

$A \cup \emptyset = A - \emptyset = A$

$A \cap \emptyset = \emptyset$

$\bar{U} = \emptyset$

$\bar{\bar{A}} = A$

$A \cup (B \cap c) = (A \cup B) \cap (A \cup c)$

$A \cap (B \cup c) = (A \cap B) \cup (A \cap c)$

$(A \cup B) = \bar{A} \cap \bar{B}$

$(A \cap B) = \bar{A} \cup \bar{B}$

حاصل ضرب دکارتی:

در برخی مواقع عناصر یک مجموعه دنباله ای از عناصر مرتب شده مجموعه های دیگر است به چنین مجموعه هایی حاصل ضرب دکارتی می گویند ؛ حاصل ضرب دکارتی دو مجموعه خود مجموعه ای است از زوج های مرتب که به صورت زیر تعریف می شوند:

$$S = s1 * s2 = \{ (x , y) | x \in s1 , y \in s2 \}$$

$S1 = \{ 2 , 4 \}$ ⟶ S1*s2=?

$S2 = \{ 2 , 5 , 6 , 7 \}$ ⟶ S1*s2=?

$S1*s2 \Longrightarrow \{ (2,2) , (2,5), (2,6), (2,7),(4, 2),(4 , 5), (4 , 6), (4,7)\}$

$S2*s1 \Longrightarrow \{ (2,2), (2,4),(5 , 2) , (5 , 4) , (6, 2) , (6, 4),(7,2),(7,4)\}$

$S1\ s2 \neq s2 * s1$

حاصل ضرب دکارتی n مجموعه

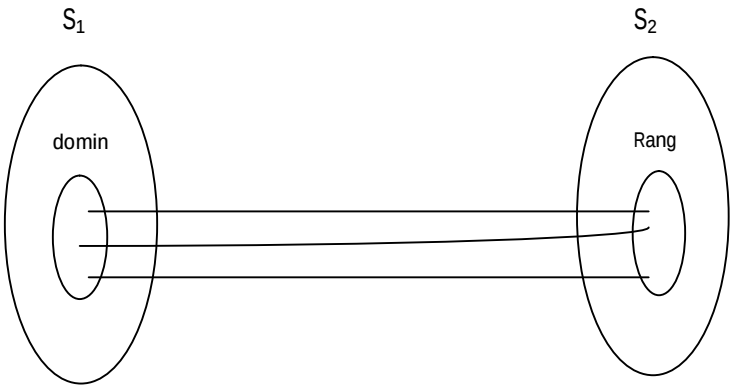
$$S^1 * s^2 * s^3 * ....*s^n = \{ (x^1,x^2,x^3,...,x^n) \mid x^1 \in s^i \}$$

تابع ضابطه ای است که عناصر یک مجموعه را به طور یکتا به مجموعه را مجموعه دیگری تخصیص می دهد ، اگر F نشان گر تابع باشد مجموعه اول دامنه یا domain و مجموعه دوم or یا Range نامیده می شود.

$F : s1 \longrightarrow s2$

$F : s1 \longrightarrow s2$

بیانگر آن است که دامنه f زیر مجموعه S و بر آن زیر مجموعه S^2 می باشد.



اگر دامنه تابع f مجموعه s1 باشد به آن تابع کلی گویند.

اگر دامنه تابع f مجموعه S باشد آن را تابع کلی (total function) گویند و امر غیر این صورت (partial function) گویند.

بسیاری از توابع را می توان به صورت مجموعه ای از زوج مرتب به شکل زیر نشان داد.

$$F = \{ (x_1 , y_1) (x_2 , y_2) \dots \}$$

که در آن x_i عنصری از دامنه و y_i عنصر متناظر آن بدبورد است.

برای این که چنین مجموعه ای تابع باشد باید هر یک از عناصر x_i فقط یکبار در طرف اول زوج ها ظاهر شود اگر اینگونه نباشد مجموعه فوق یک رابطه (Relation) است.

روابط کلی تر از توابع هستند . در یک تابع هر عنصر دامنه فقط دارای یک عنصر متناظر در برد است. اما در یک رابطه ممکن است یکه عنصر دامنه چندین عنصر متناظر در بر داشته باشد.

یک نوع رابطه مهم رابطه هم ارزی است که تعمیم از مفهوم تساوی است ، زمانی زوج x,y دارای رابطه هم ارزی هستند که با سه خاصیت زیر بر قرار باشد.

$X \equiv y$

1) $\forall x \quad x \equiv x$ بازتابی
2) $If \ x \equiv y \ then \ y \equiv x$ تقارن
3) $If \ (x \equiv y) \ and \ (y \equiv z) \ then \ (x \equiv z)$ تریایی

هم نهشتی : اگر هم نهشتی رابطه هم ارزی دارد.

$1 \overset{3}{\equiv} 4 \overset{3}{\equiv} 7 \overset{3}{\equiv} 10$

$2 \overset{3}{\equiv} 5 \overset{3}{\equiv} 8 \overset{3}{\equiv} 11$

$0 \overset{3}{\equiv} 3 \overset{3}{\equiv} 6 \overset{3}{\equiv} 6 \overset{3}{\equiv} 9 \overset{3}{\equiv} 12$

$X \overset{3}{\equiv} y \ if \quad x \mod 3 = y \mod 3$

1) $\forall x \quad x \equiv x$ بازتابی
2) $If \ x \overset{3}{\equiv} y \ then \ y \overset{3}{\equiv} x$ هم ارزی
3) $If \ x \overset{3}{\equiv} y \ and \ y \overset{3}{\equiv} 7 \Rightarrow x \overset{3}{\equiv} 7$

سه مفهوم اساسی : زبان ها – گرامر ها – آتاماتا (ماشین ها)

سه مفهوم زبان ها ، گرامرها ، آتاماتا . سه مفهوم اساسی در تئوری محاسبات هستند که به تعریف آن ها می پردازیم.

زبان :

الفبای نظریه زبان های و ماشین ها (alphabet) : مجموعه ای متناهی از نمادها را الفبا گوییم و با علامت $\sum$ نمایش داده می شوند.

$\sum = \{ a , b \}$

$\sum = \{ 0 , 1 \}$

$\sum = \{ 0 , 1 , 2 , 3 \}$

رشته (string) : دنباله ای محدود و متناهی از نمادهای الفبا را رشته گویند. یعنی از نشانه ها ی الفبا رشته ها ساخته می شوند.

$\sum = \{ a , b \}$ aab , abbbaaa , abba , baa , abbaa

رشته تهی (empety string) : رشته ای است که شامل هیچ نمادی نباشد و با علامت λ نشان داده می شود.

طول رشته : به تعداد عناصر تشکیل دهنده رشته طول رشته گفته می شود.

$W =$ abbba $|w|=5$

$|\lambda| = \emptyset \Longrightarrow \lambda w = w\lambda = w \qquad \forall w$

معکوس رشته دلخواه w : معکوس یک رشته دلخواه w را با w^R نمایش می دهند.

W^R

زیر رشته : یک زیر رشته بخشی از حروف پشت سر هم یک رشته است.

Abbba

اگر $w = vu$ باشد زیر رشته v را پیش وند (prefix) و u را پسوند گویند.

Abba

تکرار رشته ها : تکرار رشته w را به تعداد n بار به صورت W^n نشان داده می شود.

$\forall w \qquad w^0 = \lambda$

$W^1 = w$

$W^2 = ww \implies w = ab \implies W^2 = abab \implies W^3 = ababab$

تعریف $\sum^*$: به مجموعه کلیه رشته ها روی الفبا$\sum$ ، $\sum^*$ گفته می شود.(نا متناهی)

مجموعه $\sum^*$ شامل λ نیز می باشد. $\sum^*$ نا محدود است.

اگر الفبای $\sum$ دارای n عنصر باشد آن گاه n^k رشته ، به طول k در $\sum^*$ موجود خواهد بود. یعنی تعداد عناصر به توان طول رشته.

$\sum = \{a,b\}$ $n=2$ $k=2$ $2^2=4$ $\{a,a,a,b,b,a,b,b\}$

$N=2$ $k=3 \implies 2^3 = 8$ aaa aba aab bbb bba bab baa abbb

$\sum^+$: (به مجموعه تمام عناصر روی الفبای $\sum$ فلهای λ) $\sum^+ = \sum^* - \{\lambda\}$

اگر چه $\sum$ شفاهی است ، $w \sum^+$ و $\sum^*$ همواره نا متناهی هستند زیرا هیچ محدوده ای برای رشته ها وجود ندارد.

یک زبان زیر مجموعه ای از $\sum^*$ است و هر رشته در یک زبان دلخواه L یک جمله خوانده می شود.

عملگرهای رشته :

عملگر الحاق (conctatanation) :

$$\begin{cases} u = a_1 a_2 \ldots a_n \\ V = b_1 b_2 \ldots b_n \end{cases} \Longrightarrow uv = a_1 a_2 \ldots a_n \; b_1 b_2 \ldots b_n$$

عملگر توان (power)

$W = a_1 a_2 \ldots a_n$

$W^0 = \lambda$

$W^1 = a_1 a_2 \ldots a_n$

$W^2 = a_1 a_2 \ldots a_n \; a_1 a_2 \ldots a_n$

$W^m = a_1 a_2 \ldots a_n \; a_1 a_2 \ldots a_n \; a_1 a_2 \ldots a_n$

عملگر معکوس (بالیندروم):

به رشته ای بالیندرم گویند اگر و فقط اگر $w = w^R$ باشد خودش با واران آن برابر باشد.

$W = w^R$

$W = aabaa$

$w^R = aabaa \Longrightarrow W = w^R$

اگر رشته ای که از n تا w کنار هم تشکیل شود ،اگر این رشته پالیندرم باشد آن گاه w پالیندرم خواهد بود (قضیه تجزیه)

www ... ww

- اگر w ای پالیندروم باشد آن گاه (www ... w) نیز یک پالیندروم خواهد بود.

تعریف زبان : زبان به مجموعه متناهی و یا نا متناهی از رشته ها که بر روی یک الفبا تعریف شده اند زبان می گویند. و معمولا زبان را با L نمایش می دهند.

$\sum = \{ a, b, c \}$

$L_1 = \{ abc, cbb, cbac \}$ متناهی

$L_2 = \{ a (bc)^n \mid n \geq \emptyset \}$ نا متناهی

عملگر روی زبان

از آنجا که عملگرهای گفته شده در قبل روی مجموعه های متناهی و نا متناهی عمل می کنند و زبان ها نیز هم مجموعه هستند بنابراین این عملگرها روی زبان ها نیز کار می کنند.

عملگر اجتماع (Uniun):

$L_1 \cup L_2 = \{ x \mid x \in L_1 \text{ or } x \in L_2 \}$

$L_1 = \{w \mid$ رشته های w روی دنباله $\sum$ که با a شروع می شود. $\}$

$L_2 = \{ w \mid$ رشته های w روی دنباله $\sum$ که با a خاتمه می یابند. $\}$

$L_1 \cup L_2 = \{$ دنباله ای روی $\sum$ که با w شروع یا به a ختم می شوند. $\}$

اشتراک intersection :

$L_1 \cap L_2 = \{w \mid x \in L_1 \text{ and } x \in L_2\}$

- X هایی که هم عضو L_1 و هم عضو L_2 اند.

الحاق (اتصال coctaonation) دو زبان :

$L_1 . L_2 = \{ xy \mid x \in L_1 \text{ and } y \in L_2 \}$

در زبان جدید x ها از زبان اول و y ها از زبان دوم بر داشت می شوند.

$L_1 = \{ aab , aab , ab \}$

$L_2 = \{ ab, abb, aabb \}$

$L_1.L_2 = \{ aabab , aababb , aabaabb , aabab , aababb , aabaabb , abab , ababb , abaabb \}$

توان (L^n) : به معنای اتصال L به خودش به تعداد n با راست.

$L^0 = \{ \lambda \}$

$L^1 = \{ L \}$

$L^2 = \{ L . L \}$

$L^3 = L.L^2$

$L^4 = L.L^3$

بستار ستاره (star chusur) **عملگر بستار ستاره**.

$\sum = \{ a \} \longrightarrow L = \{ a \}$

$L^0 = \{ \lambda \}$

$L^1 = \{ a a \}$

$L^2 = \{ a a a a \}$

$L^3 = L . L^2 = \{ a a a a a a \}$

$L^* = L^0 \cup L^1 \cup L^2 \cup ...$

$L^* = \cup_{i=0}^{\infty} L^i$

$L^* = \{ \lambda , aa , aaaa , aaaaaa , ... \}$

بستار جمع (positive cluser)

$L^{+} = \bigcup_{i=0}^{\infty} L^{i} = L^{1} \cup L^{2} \cup L^{3} \ldots$

$L^{+} = L^{*} - \{\lambda\}$

$L^{*} = L^{+} + \{\lambda\}$

عملگر متمم (compement) :

$L^{-} = \sum^{*} - L$ ⟵ رشته هاي زبان L

کل رشته هاي موجود

عکس زبان (معکوس زبان)

$L^{R} = \{ w^{R} \mid w \in L \}$

عملگر تقسیم : تقسیم دو زبان بر هم

$L_1 / L_2 = \{ x \mid x y \in L_1 , y \in L_2 \}$

Xy ها عضو L_1 و y ها عضو L_2 هستند.

$L_1 = \{ aa , abab , bbab , abbb \}$

$L_2 = \{a , ab\}$

$L_1/L_2 = \{a , ab, bb\}$

$L_1 = \{ \lambda\ ab\}$

$L_2 = \{ ab \}$ ⟹ $L_1 / L_2 = \{\lambda\}$

ü اگر در دو زبان رشته کاملا مشترک وجود داشت λ تولید می شود.

$L = \{a^{n} b^{n} \mid n \geq 0\}$

$(a\ b)^{n}$ $a^{n} b^{n}$

L_2 و L_R را محاسبه نمایید:

$L^2 = \{a^n b^n a^m b^m : m,n \geq 0\}$

$L^R = \{b^n a^n \mid n \geq 0\}$

برای تعریف یک زبان دو روش موجود است :

الف) عبارات با قاعده یا عبارات منظم (Regulol Exporssion)

ب) گرامر

عبارات منظم : عبارات منظم روشی هستند برای نمایش رشته های یک زبان اگر $\sum$ الفبای داده شده باشد در آن صورت :

αα

1) تهی ، λ و هر عضو Σ یک عبارت منظم هستند. (ابتدایی)
2) اگر α و β دو عبارت با قاعده باشند آنگاه عبارات زیر نیز با قاعده هستند.

1) α^*
2) $\alpha +$
3) $\alpha . \beta$
4) $\alpha + \beta = \alpha \cup \beta$
5) $\alpha - \beta$
6) $\beta - \alpha$

3) رشته ای عبارت منظم است اگر و فقط اگر بتوان آن را از عبارات منظم ابتدایی با اعمال دفعات متناهی از قانون بالا تولید کرد.

-ثابت کنید با الفبای داده شده رشته ی زیر منظم است؟

$\sum = \{a,b,c\}$

$(a+b.c)^*(c+\emptyset)$

$\alpha = c$
$\beta = \emptyset$ $\Rightarrow$ $\alpha+\beta=(c+\emptyset)$ منظم

$\alpha=b$
$\beta=c$ $\Rightarrow$ $\alpha.\beta= bc$ منظم

$\alpha=a$
$\beta=bc$ $\Rightarrow$ $\alpha+\beta=a+bc$ منظم

$\alpha=a+bc$ $\Rightarrow$ $\alpha^*=(a+bc)^*$

$\alpha=(a+bc)^*$
$\beta=(i+\emptyset)$ $\Rightarrow$ $\alpha.\beta=(a+bc)^*.(c+\emptyset)$

برای الفبا$\sum = (a,b)$ آیا عبارت $r=(a+b)^*(a+bb)$ منظم است؟

برای عبارتی منظم است و بیانگر ختم به a یا bb می باشد و با a و b ساخته می شوند $r=(a+b)^*$ می تواند بیانگر λ باشد.

$L(r) = \{ a, bb, aa, abb, babbb, aaa, aabb, aba, abbb, baa, babb, bbabbbb, .. \}$

1) $L(r)$ بیانگر تمامی رشته هایی از ترکیب a,b است که در انتهای رشته یا a قرار دارد و یا bb
2) تمام رشته هایی که بر روی الفبای a,b ساخته می شود و به a یا bb ختم می گردد.

ü تمامی رشته های با تعداد زوج 0 و دنباله تعداد فرد b (روی الفبای $\sum=(a,b)$).

$r = \underbrace{(aa)}^{*} \underbrace{(bb)}^{*} b$ ⟹ اضافه کردن یك b براي فرد کردن تعداد b ها

⇩ یا هیچ یا توانی زوج از a

⇩ توان زوجي از b ها

$L(r) = \{a^{2n} b^{2m+1} \mid n \geq \emptyset , m \geq \emptyset \}$

ü پس تعداد نا محدودی عبارت برای یک زبان منظم وجود دارد.

دو عبارت منظم در صورتی مساوی هستند که یک زبان را نمایش دهند و عبارات منظم را می توان ساده کرد.

-عبارت منظمی بنویسید که کلیه رشته هایی که شامل صفر و یک باشند را توصیف کند.

$\sum = \{ 0,1 \} \Longrightarrow \quad \sum^{*}$

$r = (0+1)^{*}$

$r = (0 \cup 1)^{*}$

$L^0 = \lambda$

$L^1 = \{0,1\}$

$L^2 = \{00,01,10,11\}$

$L^3 = \{000,001,010,011,100,101,110,111\}$

-عبارت منظمی بنویسید که تمامی رشته های تشکیل شده از صفر و یک را که دارای دو صفر متوالی می باشند ایجاد نماید.

$r = \underbrace{(0+1)}^{*} \underbrace{00} \underbrace{(0+1)}^{*}$

مختار به انتخاب نزدیك به هر تعداد

مختار به انتخاب به هر تعداد از صفر و یك

الزاما درج دو صفر

-عبارت منظمی که فقط و فقط دارای دو صفر متوالی باشد و صفر دیگری نباشد.

$R = (1)^{*} 00 (1)^{*}$

- عبارت منظمی بنویسید که رشته های آن شامل 1 و 0 و حداقل دارای دو صفر باشد.

$R = (0+1)^* \underline{0}\ (0+1)^* \underline{0}\ (0+1)^*$

الزام وجود صفر — هر تعداد از صفر و یك

-عبارت منظمی بنویسید که رشته های آن شامل 0و1 باشند و با یک آغاز و نیز شامل دو صفر متوالی نباشند.

$R=(1 \cup 10)^+$

$L = \{1,10\}$

$L^2 = \{11,110,101,1010\}$

$L^3 = \{111,1110,1101,11010,1011,10110,10101,101010\}$

-عبارت منظمی بنویسید که رشته های آن شامل صفر و یک باشد و دارای دو صفر متوالی نباشند.

$R = (1+01)^*(0 + \lambda)$

-عبارت منظمی بنویسید که رشته های آن از a,b,c تشکیل شده باشند و در آن A قبل از b و b قبل از c بیاید.

$R=(A^*B^*C^*) \cup (C^*A^*)$

-عبارت منظمی بنویسید که از a,b تشکیل شده است و در آن زیر رشته aa حداکثر یک بار ظاهر شود. (یا aa وجود ندارد یا فقط یک بار وجود دارد)

$R = (b+ab)^*+(aa + \lambda)$

-عبارت منظمی که بر روی الفبای a,b تعریف شده و حداکثر دارای دو a باشد.

$\sum = \{a,b\}$

$B^*(a+\lambda)\ b^*\ (a+\lambda)b^*$ اولین جواب

$B^* \cup (b^*ab^*) \cup (b^*ab^*ab^*)$ دومین جواب

-عبارت منظمی بنویسید که بر روی الفبا a,b,c تعریف شده و تعداد نمادهای b یا c دو آن جمعاً 3 تا باشد.

$a^*(b+c) a^*(b+c) a^*(b+c)a^*$

-عبارت منظمی که بر روی الفبای a,b,c تشکیل شده و در آن بعد از هر b یک a ظاهر شده باشد.

$(ba U a U c)^*$

گرامر : برای بررسی یک زبان همواره به سه مکانیزم نیاز است.

1- مکانیزمی جهت تعریف زبان (الف) عبارات منظم یا با قاعده ب)گرامر)
2- مکانیزمی جهت تولید رشته های زبان
3- مکانیزمی جهت ارزیابی تعلق یا عدم تعلق رشته های یک زبان (مثلای از مکانیزم 2و3 اشتقاق است)

ب) گرامر : گرامر ابزاری است برای بیان ویژگی های الگوی رشته های یک زبان به عبارت دیگر گرام مکانیزمی است برای توصیف یک زبان که شامل چهار پارامتر است و به صورت زیر نمایش داده می شود.

$G = (\sum , V ,R ,S)$

1- $\sum$ **یا الفبا** (alphabet): $\sum$ حروف الفبای زبان است عناصر این مجموعه را عناصر پایانی یا ترمینال می گویند. حروف الفبا را به صورت قرار دادی با حروف کوچک انگلیسی نمایش می دهند.

2- V **متغیر** (variable) : مجموعه ای متناهی از متغیرها. یک مجموعه از عناصر هستند که از آن ها برای تعریف ساختار زبان استفاده می شود متغیر ها را غیر پایانی یا فان ترمینال نیز می گویند. غیر پایانی ها را با حروف بزرگ انگلیسی نمایش می دهند. montorminal

3- S **سمبل شروع** (starting symbol): فاز آغاز گر یک عضو ثابت مجموعه v است که به آن S گویند. S منشع هر گونه فعالیت در رابطه با زبان است.

4- R **قوانین جایگذاری** (Rewriting Rules) : یک مجموعه قاعده به صورت $x \longrightarrow y$ است که با استفاده از این قواعد می توان ساختار ویژگی های و زبان را تعریف کرد.

مفهوم $x \longrightarrow y$ به این معنی است که به جای x می توان y را قرار داد.

$x \longrightarrow y$

$x \in (\sum U V)^*$

$y \in (\sum U V)^*$

- اگر قانونی به صورت $\alpha \longrightarrow \lambda$ باشد را قانون λ گویند.
- برای هر زبان حداقل یک گرامر وجود دارد که آن زبان را توصیف کند.
- برای این که ثابت کنیم یک زبان توسط یک گرامر تولید می شود باید ثابت کنیم که :
  1) Wel تمام رشته ها می توانند توسط گرامر تولید شوند.
  2) هر جمله ای که توسط گرامر تولید می شود در زبان L ات.

-برای زبان $L=a^*$ بر روی الفبا $\sum = \{a\}$ یک گرامر بنویسید.

1) $\sum = \{a\}$
2) $V = \{S\}$
3) S
4) $R \begin{cases} S \longrightarrow \lambda \\ s \longrightarrow as \end{cases}$

λ	S
a	as
aa	aas
aaa	aaas

-برای زبان $L=a^+$ بر روی الفبا $\sum = \{a\}$ یک گرامر بنویسید؟

1) $\sum = \{a\}$
2) V={s}
3) S
4) R $\begin{cases} S \longrightarrow a \\ S \longrightarrow as \end{cases}$

S	a
as	aa
aas	aaa
aaas	aaaa

-برای زبان $L=a^*b^*$ یک گرامر بنویسید.(به ترکیباتی که اول ترکیبات a بعد ترکیبات b می آید.)

1) $\sum = \{a,b\}$
2) V={s}
3) S
4) R $\begin{cases} S \longrightarrow AB \\ A \longrightarrow \lambda \\ A \longrightarrow aA \end{cases} a^*$ $\begin{cases} B \longrightarrow \lambda \\ B \longrightarrow bB \end{cases} b^*$

S

AB

aAB

aaAB

aaaAB

aaaB

aaabB

aaabbB

aaabbbB

aaabbbbB

aaabbb

-برای زبان زیر یک گرامر بنویسید.

$L = (aa)^* bc (bb)^*$

1) $\sum = \{a,b,c\}$
2) $\{ S,A,B,C \}$
3) S
4) R

$s \longrightarrow ABC$

$A \longrightarrow \lambda$ ⟵ به دلیل داشتن * یهλ هم داریم.

$A \longrightarrow aaA$ $\}$ $(aa)^*$

$B \longrightarrow bc$

$C \longrightarrow \lambda$

$C \longrightarrow bbc$ $\}$ $(bb)^*$

$R = \begin{cases} S \longrightarrow Ab,c \\ A \longrightarrow \lambda \\ A \longrightarrow aa\,A \end{cases}$ $(aa)^*$

$\begin{cases} E \longrightarrow \lambda \\ C \longrightarrow bbc \end{cases}$ $(bb)^*$

AbcC

aaAbcc

aabcC

aabc bbC

aabc bbbbc
aabcbbbb

-گرامری بنویسید که زبان زیر را توصیف نماید.

$L=a^nb^n \qquad n\geq 0$

1) $\sum = \{a,b,c\}$
2) $\{s\}$
3) S
4) R $\begin{cases} S \longrightarrow \lambda \\ S \longrightarrow asb \end{cases}$

S

aSb

aaSbb

aabb

-گرامری بنویسید که زبان زیر را تولید کند.

$\sum = \{a,b,c\}$

$L=\{w \mid length\ (w)=3\alpha\ ,\ \alpha \in N\}$

$R=(\ (avbuc)(avbc)(avbvc)\ )^*$

$\sum = \{\ a,\ b,\ c\}$

$V=\{\ A,s\}$

$$R = \begin{cases} s \longrightarrow \lambda \\ S \longrightarrow AAAS \\ A \longrightarrow a \\ A \longrightarrow b \\ A \longrightarrow c \end{cases} \quad A \longrightarrow a|b|c$$

S

AAAS

babS

babAAAS

babaccS

-گرامری بنویسید که زبان رو برو را توصیف نماید.

$L=WW^R$

$\sum = \{a,b,c\}$

$V=\{s\}$

$S \longrightarrow aSa \mid bsb \mid csc \mid aa \mid cc$

S

aSa

abSba

abcScba

abcaSacba

-برای زبان زیر یک گرامر بنویسید:

$\sum = \{ a, b, c, d \}$

$L= a^n b^n c^m d^m \qquad n, m \geq 0$

$V=\{ s, A, B\}$

$S \longrightarrow \lambda$

$S \longrightarrow AB$

$A \longrightarrow aAb \mid \lambda \longleftrightarrow$ هرجا a تولید شود b نیز تولید می شود.

$B \longrightarrow cBd \mid \lambda \longleftrightarrow$ هرجا c تولید شود d نیز تولید می شود.

جواب دوم

$S \longrightarrow \lambda$

$S \longrightarrow AB$

$A \longrightarrow aAb|ab|\lambda$

$B \longrightarrow cBd|cd|\lambda$

-برای زبان زیر یک گرامر بنویسید:

$L = 0(10)^*$

$\sum = \{0,1\}$

$r=\{s,A\}$

$$R=\begin{cases} S \longrightarrow 0|0A \\ A \longrightarrow 10A|10 \end{cases}$$

طبقه بندی

چامسکی : در این طبقه بندی زبان ها بر اساس پیچیدگی ساختاریشان طبقه بندی شده اند هر طبقه ساختار گرامری مخصوص به خود دارد و چامسکی زبان ها را در 4 طبقه دسته بندی کرده است.

1- زبان های با قاعده یا منظم (Regular languages) زبان های نوع سوم
2- زبان های مستقل از متن (context free langueges) زبان های نوع دوم
3- زبان های وابسته به متن(context sensitive lang) زبان های نوع اول
4- زبان های برون محدودیت (unrestriceted langueges)

زبان های با قاعده (منظم)

تعریف مقدماتی:

تعریف گرامر خطی: گرامر خطی گرامری است که در آن حداکثر یک متغیر می تواند درست راست یک قانون یافت شود بدون این که محدودیتی در محل این متغیر وجود داشته باشد.

A ⟶ aaBbc

A ⟶ aabcBaa

A ⟶ Baabc خطي چپ

A ⟶ aabc abbaB خطي راست

به گرامری که متغیر آخرین حرف باشد گرامر خطی راست می گویند و به گرامری که اولین حرف باشد گرامر خطی چپ می گویند.

خود بازگشت : به قاعده ای که درست راست حداکثر یک متغیر و خود متغیر درست چپ خود بازگشت گویند.

A ⟶ Aaa خود بازگشت چپ

A ⟶ aaA خود بازگشت راست

گرامر منظم (چپ و راست): به قاعده ای به صورت A ⟶ aB یا A ⟶ Ba قاعده منظم می گویند.
در واقع هر گرامر منظمی خطی است ولی هر گرامر خطی منظم نیست.

A ⟶ abaaB
z
A ⟶ aZ خطي راست نامنظم
Z ⟶ bw

W ⟶ aL
L=aB منظم

گرامر منظم : گرامریست که مجموعه قوانین آن به شکل زیر باشد.

$$\begin{cases} A \longrightarrow \lambda \\ A \longrightarrow a \\ A \longrightarrow aB \end{cases} \qquad \boxed{\begin{array}{l} A \in \sum \\ A,B \in V \end{array}} \qquad \text{یا} \qquad \begin{cases} A \longrightarrow \lambda \\ A \longrightarrow a \\ A \longrightarrow Ba \end{cases}$$

If $\lambda \in L$

-زبان $L=a^*$ و$\sum = \{a\}$ برای این شرح گرامر منظمی بنویسید.

$$R = \begin{cases} s \longrightarrow \lambda \\ S \longrightarrow aS \end{cases}$$

$L=a^*b^*$ $\qquad \sum = \{a,b\}$

$$R = \begin{cases} s \longrightarrow \lambda \\ S \longrightarrow aS \\ S \longrightarrow bA \\ A \longrightarrow b|bA|\lambda \end{cases}$$

-زبان $L=(bUa)^*aa$ $\qquad \sum=\{a,b\}$ ؟

$$\begin{cases} S \longrightarrow aA \\ S \longrightarrow aS \\ S \longrightarrow bS \\ A \longrightarrow a \end{cases}$$

جواب خلاصه تر ⇨ $\begin{cases} s \longrightarrow as|bs|aA \\ A \longrightarrow a \end{cases}$ تمام ترکیبات a,b

$L = b^*(ab^* ab^* ab^*)^*$

$S \longrightarrow \lambda$

$S \longrightarrow bs|aA$

$A \longrightarrow bA|aB$

$B \longrightarrow bB|as$

-برای زبان زیر گرامر منظم تولید کنید.

$L = (a \cup b \cup c)^3$

$S \longrightarrow aA|bA|cA$

$A \longrightarrow aB|bB|cB$

$B \longrightarrow a|b|c$

زبان های مستقل از متن

زبان L را مستقل متن اگر برای آن یک گرامر مستقل از متن وجود داشته باشد. به گرامری مستقل از متن گویند که قوانین آن به صورت زیر باشند.

$$\begin{cases} S \longrightarrow \lambda & \text{if } \lambda \in L \\ A \longrightarrow X \begin{cases} A \in V \\ X \in (\sum UV)^* \end{cases} \end{cases}$$

در گرامر منظم از هر دو طرف قانون محدودیت وجود دارد و اما در زبان های مستقل از متن سمت راست کمی قوانین آزادتر شده است.

- هر گرامر منظم مستقل از متن است بنابراین هر زبان منظم مستقل از متن نیز می باشد اما عکس آن برقرار نیست.

ü هر گرامر خطی مستقل از متن است اما هر گرامر مستقل از متنی خطی نیست.

گرامر مستقل از متن بنویسید.

$L = (b \cup a)^* aa$

$L=aa\overline{V (b \cup a)}^{+} \overline{aa}^{B}$

$S \longrightarrow aa|AB$

$B \longrightarrow aa$

$A \longrightarrow aA|bA|a|b$

-برای زبان $L=ww^R$ یک گرامر مستقل از متن بنویسید.

$L=ww^R$

$S \longrightarrow aSa\ |bSb$

$S \longrightarrow aa|bb$

-برای زبان زیر $L=a^nb^nc^md^m$ یک گرامر مستقل از متن بنویسید.

$\lambda \begin{cases} n = 0 \\ m = 0 \end{cases}$ $\quad \underbrace{c^m d^m}_{B} \begin{cases} n = 0 \\ m > 0 \end{cases}$ $\quad m, n \geq 0$

$\underbrace{a^n b^n}_{A} \begin{cases} n>0 \\ m=0 \end{cases}$ $\quad \underbrace{a^n b^n}_{A} \underbrace{c^m d^m}_{B} \begin{cases} m>0 \\ n>0 \end{cases}$

$s \longrightarrow \lambda|B|A|AB$

$A \longrightarrow aAb|ab$

$B \longrightarrow CBd|cd$

-برای زبان $L=a^n b^m c^{n+m}$ یک گرامر مستقل از متن بنویسید.

$\frac{\lambda}{n=0}$ $\frac{b^m c^m}{n=0}$ $\frac{a^n c^n}{n>0}$ $\frac{a^n b^m c^{n+m}}{n>0}$ n ,m > Ø

M=0 m>0 (B) m=0 (A) m>0 (Z)

S ⟶ λ|A|B|Z

A ⟶ aAc|ac

B ⟶ bBc|bc

z ⟶ aZc|aBc

به تعداد c،a نیز تولید می کند.

-یک گرامر مستقل از متن برای زبان $L=a^n b^m c^{n-m}$ بنویسید.

$n \geq m \geq 0$

$\frac{\lambda}{m=0}$ $\frac{a^n c^n}{m=0}$ $a^n b^m c^{n-m}$

n=0 n>0 n≥ m > Ø

W1 m=n

⟹ If i=Ø ⟹ $a^m b^m$

W2 if i ≥0⟹ $a^i a^m b^m c^i$

⟹

n > m

i =n –m

n =i+m

$L = a^{i+m} b^m c^i$

حالت سوم را می توان به این شکل تبدیل کرد:

$a^i a^m b^m c^i$

$S \longrightarrow \lambda|A|Z$

$A \longrightarrow aAc|ac$

$Z \longrightarrow W_1|W_2$

$W_1 \longrightarrow aw_1b|ab$

$W_2 \longrightarrow aw_2c|aw_1c$

-گرامر مستقل از متن بنویسید.

$L = a^n b^m \quad n \geq m \geq 0$

$\frac{\lambda}{m=0} \quad \frac{a^n}{m=0} \quad a^n b^m$

$n=0$ $\quad$ $n>0$ $\quad$ $n \geq m > 0$ ⟨ 1) $n = m > 0$ $a^n b^n$; 2) $n > m > 0$ $a^n b^m$

$S \longrightarrow \lambda|A|B$

$A \longrightarrow aA|a$

$B \longrightarrow aBb|ab|aAb$

تغیر متغیر $a>b$

زبان های وابسته به متن : یک زبان را در صورتی وابسته به متن می گویند اگر بتوان برای آن زبان یک گرامر وابسته به متن نوشت. گرامر وابسته به متن دارای قوانینی به شکل زیر می باشد.

$\alpha \longrightarrow \beta$

$\alpha , \beta \in (\sum U V)^+$ نه در سمت راست λداریم و نه در سمت چپ.

$length\ (\alpha) \leq length\ (\beta)$ محدودیت (انبساطی)

$|\alpha| \leq |\beta|$

زبان های وابسته به متن جمله λ را تولید نمی کنند ، هر گرامر مستقل از متنی که جمله λ را تولید نکند یک گرامر وابسته به متن خواهد بود.

زبان های وابسته به متن خاصیت غیر انقباظی دارند یعنی طول جمله های پشت سر هم هر گز نمی تواند کوتاه تر شود.

$L = \{ a^n b^n c^n \quad n \geq 1\}$

$S \longrightarrow abc \mid aAbc$

$Ab \longrightarrow bA$

$Ac \longrightarrow Bbcc$

$bB \longrightarrow Bb$

$aB \longrightarrow aa \mid aaA$

زبان های بدون محدودیت : هر زبانی که بتوان یک گرامر بدون محدودیت برای آن نوشت یک زبان بدون محدودیت خواهد بود

$\alpha \longrightarrow \beta$

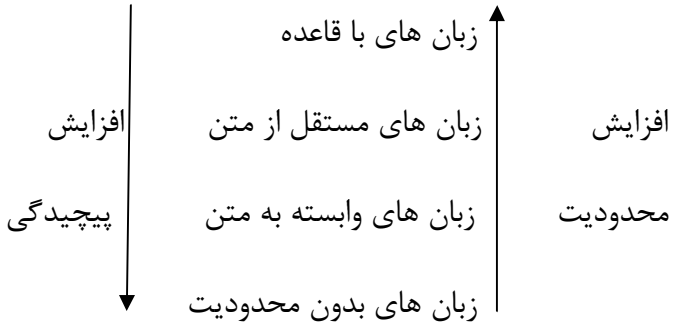
$\alpha \in (\sum U V)^+$

$\beta t(\sum UV)^*$

زبان های بدون محدودیت ⊂ زبان های مستقل از متن ⊂ زبان با قاعده

زبان های بدون محدودیت ⊂ زبان های وابسته به متن

زبان های وابسته به متن ⊂ زبان های مستقل از متن بدون λ



برای آن که ثابت کنیم زبان L در طبقه K قرار دارد باید برای آن زبان یک گرامر در طبقه K ارائه نماییم.

اشتقاق (drivation) **مکانیزمی جهت** :

الف) تولید رشته های زبان

ب) ارزیابی تعلق یا عدم تعلق رشته ها به یک زبان است.

$G = (V , \sum , S , R)$

$L (G) = \{ w \in \sum^{*} : S \overset{\star}{\Longrightarrow} W \}$ اگر یک گرامر باشد.

پس آنگاه $L(G)$ با این تعریف زبان تولید شده توسط G است.

اگر $W \in L (G)$ باشد آنگاه $S \Longrightarrow W1 \Longrightarrow W2 \ldots\ldots \Longrightarrow W$ را یک اشتقاق از جمله W می گویند. پس رشته های $s,w1,w2,\ldots$ را که حاوی ترمینال های و متغیر ها هستند شکل های گذاره ای (sehtencidform) می گویند. پس به طور خلاصه : اشتقاق مکانیزمی است جهت تولید جملات از گرامر یا به عبارتی دیگر به دنباله ای از قوانین جایگزینی که منجر به تولید جمله از گرامر می شود اشتقاق می گویند.

تعریف شبه جمله : به عبارت های موجود در اشتقاق که حاوی ترمینال و غیر ترمینال است شبه جمله می گویند.

جمله : شبه جمله ای که تمام محتوای آن ترمینال یا الفبا باشد جمله گویند.

مثال)

$V = \{s\}$ $\sum=\{a,b\}$ $R:\{s \longrightarrow aSb|\lambda\}$

$S \rightarrow asb \longrightarrow aasbb \rightarrow aabb$

$S \overset{☆}{\rightarrow} aabb$

با توجه به این که انتخاب قواعد در زبان انجام عمل اشتقاق تاثیری در نتیجه نهایی نخواهد داشت لذا باید انتخاب قواعد را از حالت سلیقه ای خارج کرد و بر اساس معیار معلومی قواعد را انتخاب و بر اساس آن متغیر مورد نظر بسط داده شود.

انواع اشتقاق : 1- اشتقاق چپ

اشتقاق چپ : در این اشتقاق در هر بار سمت چپ ترین متغیر موجود در شبه جمله جایگزین می شود.

اشتقاق راست : در این اشتقاق هر بار سمت راست ترین متغیر موجود در شبه جمله استفاده می شود.

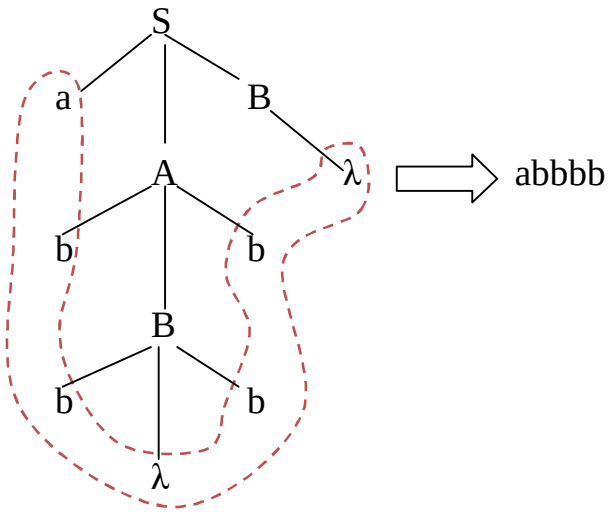
در گرامر روبرو جمله abbbb را تولید نمایید.

$$\begin{cases} S \longrightarrow aAB \\ A \longrightarrow bBb \\ B \longrightarrow A|\lambda \end{cases}$$

چپ : $S \longrightarrow aAB \longrightarrow abBbB \rightarrow abAbB \rightarrow abbBbbB \rightarrow abbbbB \rightarrow abbbb$

راست : $S \rightarrow aAB \rightarrow aAA \rightarrow aAbBb \rightarrow aAbb \rightarrow abBbbb \longrightarrow abbbb$

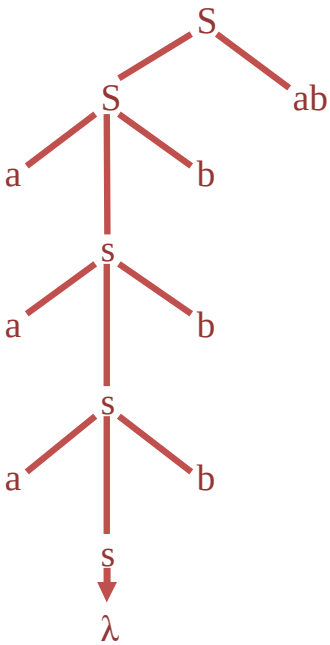
درخت اشتقاق : درختی است جهت نمایش مراحل اشتقاق ، در درخت اشتقاق نماد آغازگر (S) به عنوان ریشه و گره های داخلی به عنوان متغیر و گره های خارجی (برگ) به عنوان اعضای الفبا یا همان ترمینال محسوب می شوند.

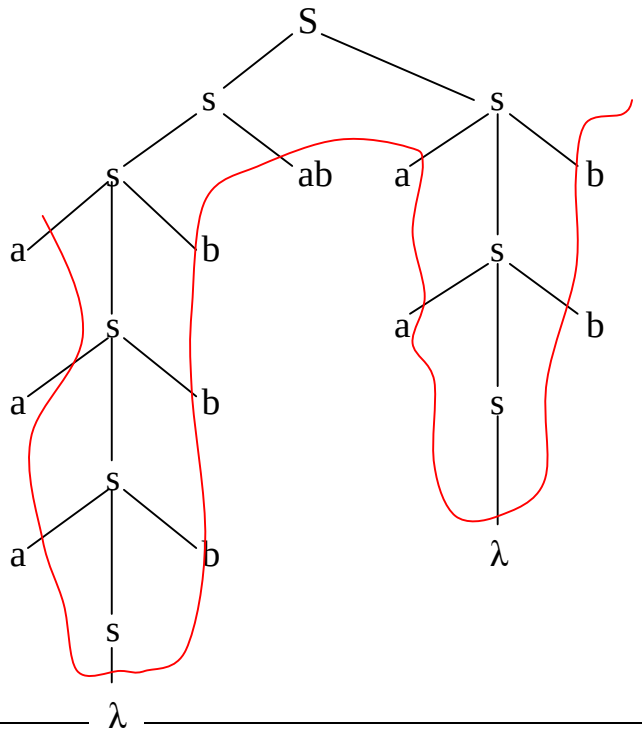


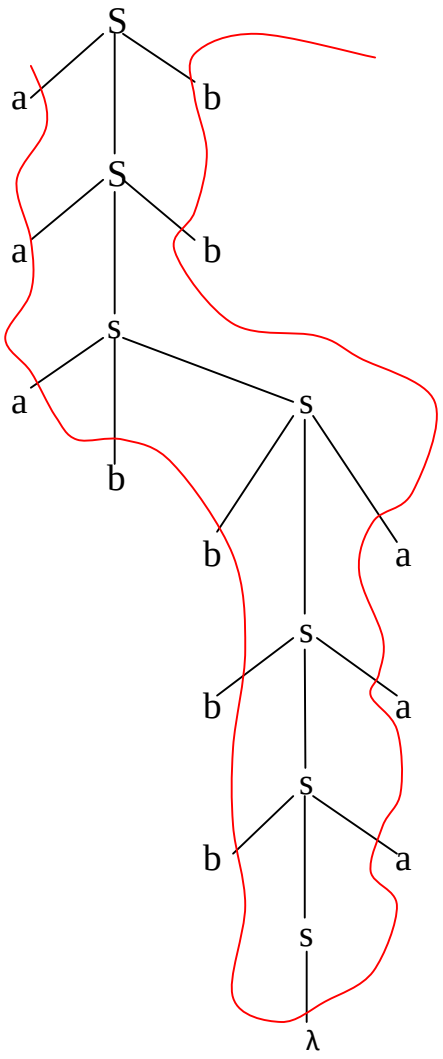
$L = \{ w \mid w \in \{ a, b\} \text{ and } Na=Nb\}$

$S \longrightarrow aSb|bsa|abs|bas|sba|\lambda|ss$

درخت اشتقاق برای تولید رشته aaabb ba baabb







aaabbbabaabb

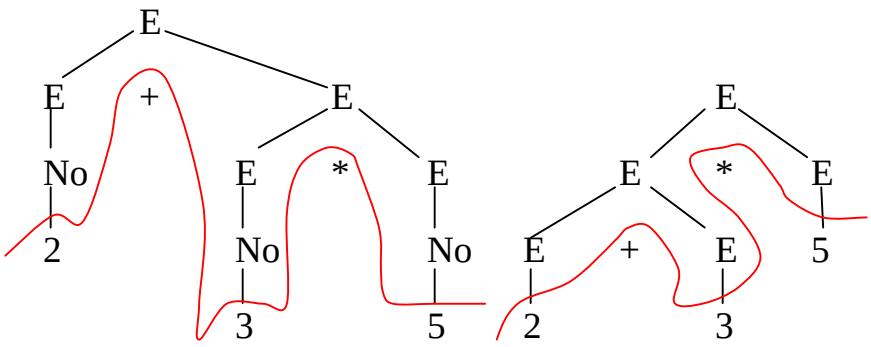
از آنجا که هر درخت اشتقاق یک تحلیل معنایی برای مورد بررسی ارائه می کند پس وجود دو درخت اشتقاق متفاوت به ازای یک رشته متعلق به زبان به معنای مبهم بودن آن زبان یا گرامر است. در واقع به گرامری که دارای بیش از یک درخت اشتقاق حداقل برای تولید یک جمله باشد گرامر مبهم می گویند.

E ⟶ E+E

E ⟶ E*E

E ⟶ Id | No

W=2+3*5



2+(3*5)=17 (2+3)*5 = 25

تمام گرامرهایی که برای یک گرامر دو درخت اشتقاق داشته باشند مبهم اند.

پس به طور کلی مبهم بودن یک گرامر را فقط با رسم دو درخت اشتقاق متفاوت به ازای یک رشته متعلق به زبان می توان تشخیص داد ، اما اگر در یک گرامر یکی از شرایط زیر مشاهده شد می توان نسبت به مبهم بودن آن گرامر شک کرد:

1) **وجود متغیر بازگشتی** : به متغیری که بعد از یک و یا چند بار اشتقاق خودش را تولید نماید متغیر بازگشتی گویند.

S ⟶ abAaa|λ

A ⟶ aB|b

B ⟶ bs|bba

متغیر بازگشتی مستقیم : متغیری که بعد از یک بار استفاده از دستور جایگزینی خودش را تولید نماید.

A ⟶ abAa|λ

A ⟶ aaA راست

A ⟶ Aaba چپ

متغیر بازگشتی غیر مستقیم : به متغیربازگشتی که بعد از چند بار اشتقاق دوباره به وجود آید متغیر بازگشتی غیر مستقیم گویند. (مانند S در مثال بالا)

متغیر λ : به متغیری که λ را تولید نماید متغیر لاندا گفته و به قاعده آن قاعده لاندا گویند.

متغیر لاندا ⟹ A ⟶ λ

2) **زنجیر** : به قوانینی مانند A ⟶ B, B ⟶ C را زنجیر گویند.

3) **متغیر غیر مفید** : به متغیری که نقشی در تولید جملات زبان نداشته باشد متغیر غیر مفید گویند و در حالت موجودات 1) از S به آن متغیر نرسیم.

2) قواعدی که به جای آن متغیر فقط ترمینال قرار بگیرد وجود نداشته باشد.

S ⟶ abacS|aA|λ

A ⟶ bbA

B ⟶ bB|b ← غیر مفید

S ⟶ Aaab|bB

A ⟶ aA|a

B ⟶ bBb

S

bB

bbBb

bbbBbb

B متغیر غیر مفید

شکیات مهم : 1- وجود قواعد بازگشت چپ و راست به ازای یک متغیر به صورت همزمان

$A \longrightarrow aA|Aa$

$S \longrightarrow SaSb|bSas$

2- وجود قواعد خود بازگشت چپ یا راست که به همراه آن متغیر بازگشتی از وسط یک قاعده ظاهر شده باشد.

$A \longrightarrow aA|aAb$

$A \longrightarrow Aa|aAb$

3- اگر بتوان از یک متغیر با تکرار فرایند جایگذاری مجددا به همان متغیر رسید.

$A \longrightarrow \ldots\alpha \longrightarrow \ldots \longrightarrow A$

$S \longrightarrow aS|bS|A|\lambda$

$A \longrightarrow aA|bA|s|\lambda$

مثال) آیا گرامر های زیر مبهم هستند.

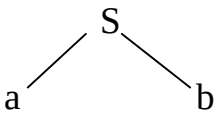
$S \longrightarrow aSA|ab|\lambda$

$A \longrightarrow bA|\lambda$

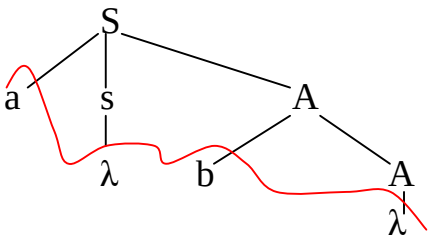
- وجود قواعد خود بازگشت در خط دوم.

یک رشته ساده را جهت ترسیم درخت انتخاب می کنیم.

ab



ü ab



§ Ab

گرامرهای نرمال : گرامرهای نرمال گرامرهای مستقل از متنی می باشند که در آن ها قوانینی که باعث ابهام در گرامر می شوند وجود ندارد . دو گرامر نرمال معروف عبارتند از 1 - **گرامر نرمال چامسکی**

2 - **گرامر نرمال گری پاخ**

فرم نرمال چامسکی : گرامر چامسکی گرامری می باشد که قوانین آن به اشکال زیر می باشد.

1) $S \longrightarrow \lambda$ if $\lambda \in L(G)$
2) $A \longrightarrow a$ $A \in V, a \in \sum$
3) $A \longrightarrow A_1A_2$ $A, A_1, A_2 \in V$

مراحل تبدیل یک گرامر به گرامر نرمال چامسکی :

1- حذف نماد آغازگر بازگشتی
2- حذف متغیر های λ
3- حذف زنجیرها
4- حذف متغیرهای غیر مفید
5- حذف بازگشتی چپ
6- تبدیل نهایی

1) **حذف نماد آغازگر بازگشتی** (s): نماد آغازگر باید تنها آغازکننده فرایند اشتقاق باشد بنابراین ظاهر شدن نماد آغازگر در مراحل اشتقاق با تعریف آن در نماد است. ولی در صورتی که در گرامر $G=(\sum,V,R,S)$ نماد آغازگر (S) بازگشتی باشد آنگاه گرامر $G`$ وجود خواهد داشت که در آن نماد آغازگر s غیر بازگشتی است و $L(G)=L(G`)$ است.

$G = (\sum ,V, R, S)$

$G` = (\sum ,V, R, S)$

$S \longrightarrow aSb|ab \Longrightarrow \begin{cases} s` \longrightarrow s \\ s \longrightarrow aSb|ab \end{cases}$

2) **حذف قاعده** λ : در صورتی که زبان حاوی رشته λ باشد تنها نماد آغازگر گرامر باید آن را تولید نماید هیچ یک از گرامرهای موجود در گرامر نباید قاعده λ داشته باشند.

$a^*b^*c^*$

$s \longrightarrow ABC$

$A \longrightarrow aA|\lambda$

$B \longrightarrow bB|\lambda$

$C \longrightarrow cC|\lambda$

$S \longrightarrow \lambda$

$S \longrightarrow BC$

$S \longrightarrow AC$

$S \longrightarrow AB$

$S \longrightarrow C$

$S \longrightarrow B$

$S \longrightarrow A$

$S \longrightarrow ABC$

$A \longrightarrow aA|a$

$B \longrightarrow bB|b$

$C \longrightarrow cC|c$

3) **حذف قاعده زنجیره ای** : قاعده زنجیره ای موجب افزایش تعداد مراحل اشتقاق می شود لذا با حذف آن سرعت فرآیند افزایش پیدا می کند.

$$\begin{cases} A \longrightarrow B \\ B \longrightarrow \alpha_1|\alpha_2|\alpha_3|\ldots|\alpha_n \end{cases} \Longrightarrow \begin{cases} A \longrightarrow \alpha_1|\alpha_2|\ldots|\alpha_n \\ B \longrightarrow \alpha_1|\alpha_2|\ldots|\alpha_n \end{cases}$$

$\alpha \longrightarrow$ مجموعه رشته و گرامر

مثال)

$$\begin{cases} A \longrightarrow B \\ B \longrightarrow aBab|ab|a \end{cases} \Longrightarrow \begin{cases} A \longrightarrow aBab|ab|a \\ B \longrightarrow aBab|ab|a \end{cases}$$

4) **حذف متغیرهای غیر مفید** : یک متغیر غیر مفید متغیری است که یکی از شرایط های زیر را داشته باشد:

1- از نماد آغازگر قابل دسترس نباشد یا به بیان دیگر در زنجیره اشتقاق به ازای هیچ رشته متعلق به زبان ظاهر نگردد.

2- هیچ گاه ترمینال تولید نکند یعنی قادر به تولید پایانی نباشد.

پس از شناسایی متغیر غیر مفید کلیه قواعد حاوی متغیر غیر مفید باید به گونه ای حذف یا تغیر یابند که تاثیری در تولید رشته ها نداشته باشد.

$S \longrightarrow ABD|ABC$

$A \longrightarrow aA|a$ AB مفید

$B \longrightarrow AB|aA$

$D \longrightarrow DB|bC$ CDEF غیر مفید

$F \longrightarrow aF|C$

$E \longrightarrow aE|b$

نهایی ⇨
- $S \rightarrow AB$
- $A \rightarrow aA|a$
- $B \rightarrow AB|aA$

مثال)

$S \rightarrow AaS \mid \lambda$

$A \rightarrow Bma|aAb|Nab$

$B \rightarrow bbBa|aB$

$N \rightarrow Fas|nK$

$K \rightarrow aK|a$

S A B M N F K

B: غیر مفید

M: غیر مفید

F: غیر مفید

بهینه شده ⇨
- $S \quad AaS \mid \lambda$
- $A \quad aAb \mid Nab$
- $N \quad nK$
- $K \quad aK \mid a$

5) حذف بازگشتی چپ

$A \rightarrow A\alpha|\beta$

$A \rightarrow BD|B$

$D \rightarrow \alpha D|\alpha$

- $A \rightarrow A\alpha_1|A\alpha_2|\ldots|A\alpha_n$
- $A \rightarrow B_1|B_2|\ldots|B_n$
- $A \rightarrow B_1D|B_2D|\ldots|B_nD|B_1|B_2|\ldots|B_n$
- $D \rightarrow \alpha_1D|\alpha_2D|\ldots|\alpha_nD|\alpha_1|\alpha_2|\ldots|\alpha_n$

$$A \longrightarrow \overline{Aaba}|\overline{Aaa} \quad (\alpha 1,\ \alpha 2)$$

$$A \longrightarrow \overline{abbA}|\overline{baA}|\overline{bb} \quad (B1,\ B2,\ B3)$$

$$\begin{cases} A \longrightarrow abbAD|baAD|bbD \\ A \longrightarrow abbA|baA|bb \\ D \longrightarrow abaD|aaD \\ D \longrightarrow aba|aa \end{cases}$$

مرحله 5 : **حذف بازگشتی چپ**

$$\begin{cases} S` \longrightarrow s \\ S \longrightarrow bD|b \\ D \longrightarrow aaABD \mid aaAD \mid aaAB \mid aaA \\ A \longrightarrow bA|a \\ B \longrightarrow bB|b \end{cases}$$

حذف غیر مفیدها (4)

$$\begin{cases} S` \longrightarrow S \\ S \longrightarrow \overline{SaaAB}|\overline{SaaA}|\overline{b} \quad (\alpha 1,\ \alpha 2,\ B1) \\ A \longrightarrow bA|a \\ Ba \longrightarrow bB|b \end{cases}$$

6)**تبدیل به فرم چامسکی**:

$$S \longrightarrow bD \begin{cases} s \longrightarrow T_5D \\ T_5 \longrightarrow b \end{cases}$$

$$D \longrightarrow aaA \begin{cases} D \longrightarrow T_1A \end{cases}$$

$$D \longrightarrow aaAB \Longrightarrow \begin{cases} D \longrightarrow T_1T_2 \\ T_1 \longrightarrow T_3T_4 \\ T_2 \longrightarrow AB \\ T_3 \longrightarrow a \\ T_4 \longrightarrow a \end{cases}$$

تبدیل نهایی : در تبدیل نهایی موارد مربوط به فرم چامسکی رعایت می شود.

$A \longrightarrow a_1a_2$: $A \longrightarrow T_1T_2$, $T_1 \longrightarrow a_1$, $T_2 \longrightarrow a_2$

$A \longrightarrow A`a$ : $A \longrightarrow A`T$, $T \longrightarrow a$

$A \longrightarrow aA`$ : $A \longrightarrow TA`$, $T \longrightarrow a$

$A \longrightarrow a1a2a3$: $A \longrightarrow T_1T_2$, $T_1 \longrightarrow T_3T_4$, $T_3 \longrightarrow a_1$, $T_4 \longrightarrow a_2$, $T_2 \longrightarrow a_3$

فرم نرمال گوی باخ : به گرامر مستقل از متن نرمالی که قوانین آن به شکل زیر باشد قوانین نرمال گری باخ می گویند.

1) $S \longrightarrow \lambda$ if $\lambda \in L(G)$
2) $A \longrightarrow a$ $A \in V, a \in \sum$
3) $A \longrightarrow aA_1A_2A_3 \ldots A_n$ $A_1, A_2, A_3, \ldots \in V$

مراحل تبدیل یک گرامر ، به گرامر نرمال گری باخ همان 6 مرحله چامسکی است فقط در مرحله 6 یا همان تبدیل نهایی به جای قوانین چامسکی باید قوانین گری باخ را در نظر گرفت.

$S \longrightarrow SaaAB \mid b$

$A \longrightarrow bBD \mid bA \mid a$

$D \longrightarrow KZ \mid dD$

$Z \longrightarrow cZ \mid D$

$B \longrightarrow bB \mid \lambda$

مثال)

S ⟶ SaaAB|b	S ⟶ S`
1 S ⟶ S	S ⟶ SaaAB|b|SaaA
A ⟶ bBD|bA|a	A ⟶ bBD|bA|a|bD
2 D ⟶ KZ|dD	D ⟶ KZ|dD
Z ⟶ cZ|D	Z ⟶ Z|D
B ⟶ bB|λ	B ⟶ bB|λ

حذف متغیرهای غیر مفید Z,D

4

S ⟶ S`	S ⟶ S`
حذف زنجیر S ⟶ SaaAB | b| SaaA	S ⟶ SaaAB |b |SaaA
3 A ⟶ bBD | bA |a |bD	A ⟶ ~~bBD~~| bA| a| ~~bD~~
D ⟶ KZ|dD	~~D ⟶ KZ| dD~~
Z ⟶ cZ|KZ|dD	~~Z ⟶ cZ |kz |dD~~
B ⟶ bB|b	B ⟶ bB |b

آتاماتا (Automata) :

ماشین یک مدل انتزاعی از یک کامپیوتر است. یک ماشین از تعدادی حالت تشکیل شده است و با دریافت ورودی و پردازش آن به یکی از حالت های تعریف شده تغییر وضعیت می دهد. ماشین در صورتی که در یک حالت مقبر قرار گیرد می توان به این نتیجه رسید که رشته ورودی صحیح بوده است. رشته ورودی می تواند دنباله ای از دستورات باشد که با اجرای هر یک از دستورات ماشین تغییر حالت می دهد.

در واقع یک ماشین بررسی می کند که آیا رشته W مربوط به زبان L می باشد یا خیر به همین دلیل به آن پذیرنده (accepper) و یا تشخیص دهنده نیز می گویند.

انواع ماشین ها

1) **ماشین متناهی**(finite state Automate) FA : این ماشین پذیرنده زبان های با قاعده هستند.
2) **ماشین های پشته ای**(fush domn Automat)FDA : ماشین پذیرنده زبان های مستقل از متن هستند.
3) **ماشین های پشته ای تقویت شده** (Automrnted push doen Automata) : این ماشین ها پذیرنده ی زبان های وابسته به متن هستند.
4) **ماشین های تورینگ** (Turing machine) linur bounded machine : این ماشین ها پذیرنده زبان های بدون محدودیت هستند.

ماشین های حالت متناهی (FA) به سه دسته تقسیم می شوند:

1) DFA 2) NFA λ-NFA3

DFA : Deter ministic FA
NFA : Non-Deter ministic FA

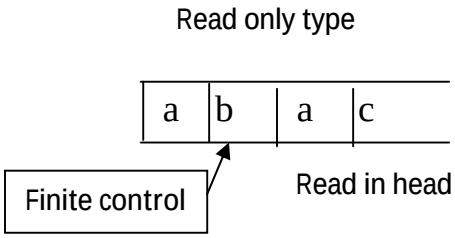
DFA **ماشین های حالت متناهی قطعی** : از سه بخش اصلی تشکیل شده است:

1) **نوار ورودی** : که به آن نوار مغناطیسی یا نوار فقط خواندنی نیز می گویند.Read only type
2) **هد نوار خوان** : (Reading Head)
3) **کنترل متناهی** : یا همان ثبات وضعیت (finite control)

1)**نوار ورودی** : نوار ورودی از سمت چپ متناهی و از سمت راست نا متناهی است ، این نوار به خانه هایی تقسیم شده است و در هر یک از خانه های نوار یکی از حروف الفبای زبان ذخیره شده است.نوار فقط خواندنی است و رشته ورودی را در خود جای می دهد.

2) **هد نوار خوان** : نوار خوان در هر لحظه مقابل یکی از خانه های نوار قرار دارد با اجرای یک دستور در ماشین نوار خوان مقداری را از ورودی هد می خواند و یک واحد به سمت راست حرکت می کند.

3) **کنترل متناهی یا ثبات وضعیت** : بخش اصلی یک ماشین است در واقع یک حافظه است که وضعیت جاری ماشین را در خود ذخیره می کند. عملکرد ماشین به این صورت است که ابتدا کنترل متناهی در حالت اولیه قرار می گیرد و سپس در هر مرحله بر اساس آنکه در چه حالتی قرار دارد و حرف خوانده شده از رشته ورودی چه چیزی باشد به حالت جدیدی می رود و در نهایت هنگامی که رشته ورودی تمام می شود در حالت ها باقی می ماند.



تعریف ریاضی (dfa) : DFA یک پنج تایی مرتب است.

$$M= (\sum ,q ,Q ,S ,F)$$

$\sum$: حروف الفبای ماشین

q: حالت شروع ماشین (حالت ابتدایی)

Q : مجموع متناهی حالات ماشین

S : مجموعه دستورات ماشین (قوانین)

F : مجموعه حالات نهایی (پذیرنده ماشین)

$q_0 \in Q$

$F \in Q$

$S : Q^{*}\sum \longrightarrow Q$

شرط پذیرش : در ماشین های قطعی هد نوارخوان نمی تواند به سمت چپ حرکت کند بنابراین آنچه در زبان های قبل خوانده شده قبلا در حالت های ماشین تاثیر گذاشته است و دیگر نمی تواند تاثیر ایجاد کند.

پیکر بندی dfa را که وضعیت آن و رفتار آتی آن را نمایش می دهد با (q,w) نمایش می دهند که در آن q حالت فعلی و W رشته در حال پردازش است.

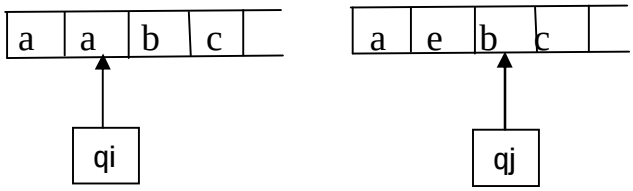
شرط پذیرش در ماشین متناهی قطعی آن است که هنگامی که تمام رشته ورودی پردازش شد حالت ماشین یکی از حالت های نهایی باشد به عبارت دیگر رشته w توسط ماشین M پذیرفته می شود اگر و فقط اگر حالتی مانند $q \in f$ وجود داشته باشد به طوری که $(q,w) \overset{*}{\underset{m}{|}} (s,w)$ در واقع پس از اتمام خواند رشته w توسط ماشین و رسیدن به λ در صورتی که حالت q جز حالت پایانی باشد رشته پذیرفته شده است ولی اگر q جز حالات پایانی نباشد رشته پذیرفته نیست.

یک دستورالعمل در ماشین به صورت زیر تعریف می شود:

$S(Q,\sum) \longrightarrow q_i$

$S(q_i,a)=q_j$

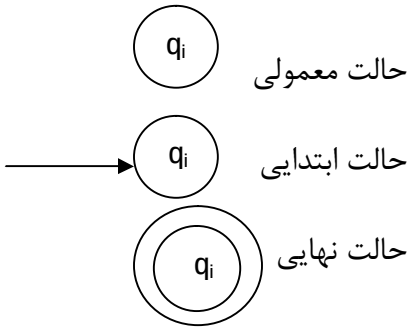
$S(q_j,b)=q_K$



تفاوت بین ماشین های DFA و NFA : ماشین های dfa در تصمیم گیری خود در زمان پردازش ورودی دارای قطعیت هستند،برعکس در ماشین های nfa قطعیتی در عبور از یک وضعیت به وضعیت دیگر وجود ندارد و در هر حالت با ورودی ثابت می توان به یک و یا چند حالت دیگر تغییر وضعیت داد.

دیاگرام حالت (گراف تغییر حالت): برای آنکه بتوان عملکرد ماشین را به صورت قابل درک نمایش داد از ساختاری به نام گراف یا دیاگرام گذر حالت استفاده می شود و دیاگرام : حالت ماشین گرافی جهت دار می باشد که در آن گره ها حالات ماشین و لبه ها یا همان یالها نشان دهنده ی الفبای ورودی می باشند و هر دستورالعمل به صورت $s(q_i, a) = q_j$ با نماد زیر نمایش داده می شود.

در گراف جهت دار سه حالت داریم



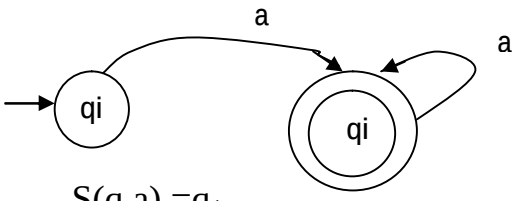
نکته : هما نطور که قبلا گفته شد شرط پذیرش یک رشته توسط DFA

1- توقف ماشین در حالت نهایی

2- خاتمه ورودی

مثال) برای زبان $L=a^+$ یک Dfa طراحی کنید.

$\sum=\{a\}$



1- $\sum=\{a\}$

2- $Q=\{q_0,q_1\}$

3- $F=\{q_1\}$

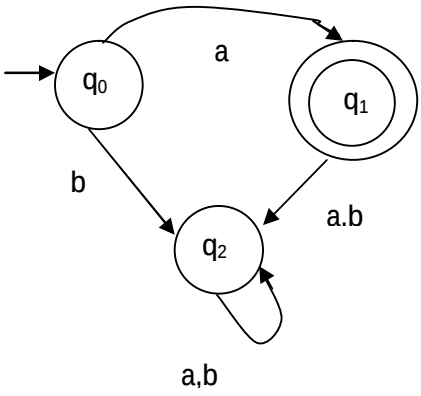
4- $S=(q_1,a)=q_1$ $S=s(q_0,a)=q_1$ $S(q,a)=q_1$ $S(q,a)=q_1$ $S(q,a) =q_1$

5- q_0

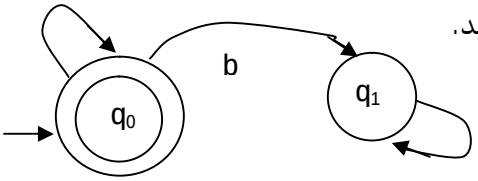
نکته : ماشین های DFA توانایی بحث در مورد پذیرش رشته های معتبر و همچنین عدم پذیرش رشته های نامعتبر را دارند.

مثال) برای زبان $L = \{a\}$ یک DFA طراحی کنید:

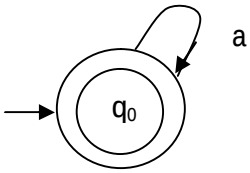
$\sum=\{ a, b\}$



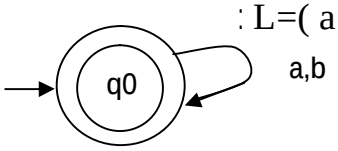
مثال) برای زبان $L=a^*$ بر روی $\sum=\{a,b\}$ یک DFA طراحی کنید.



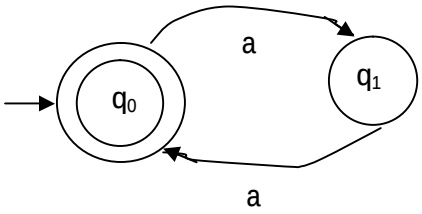
مثال) برای زبان $L=a^*$ و الفبای $\sum=\{a\}$ یک DFA طراحی کنید.



-برای زبان $L=(a+b)^*$:

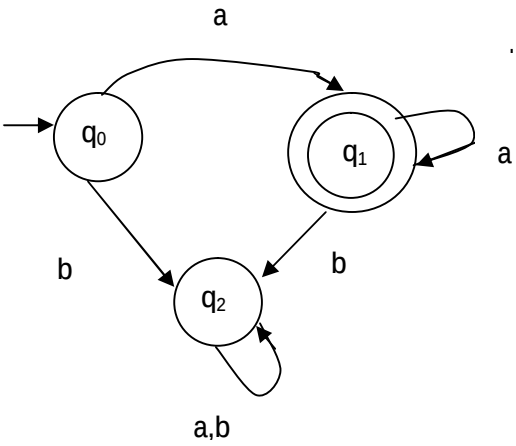


مثال) برای زبان $L=(aa)^*$ یک DFA طراحی نمایید.

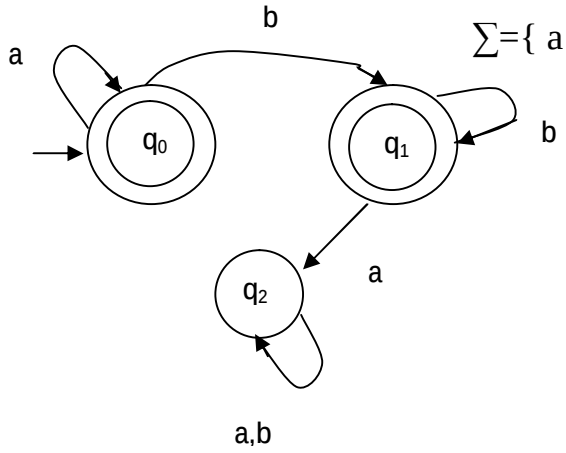


$\sum=\{A\}$ $L=(aa)^*$

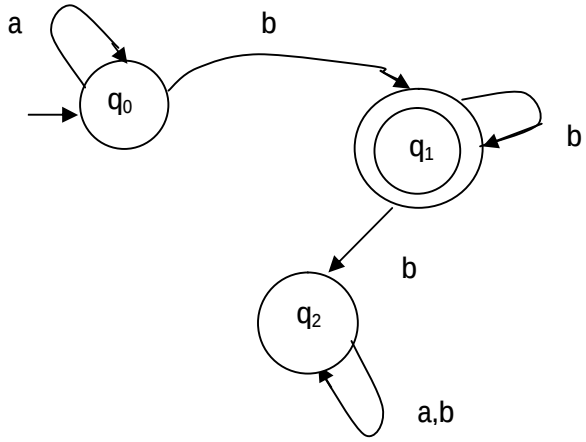
مثال) برای زبان $L=a^+$ بر روی $\sum=\{a,b\}$ یک dfa رسم کنید.



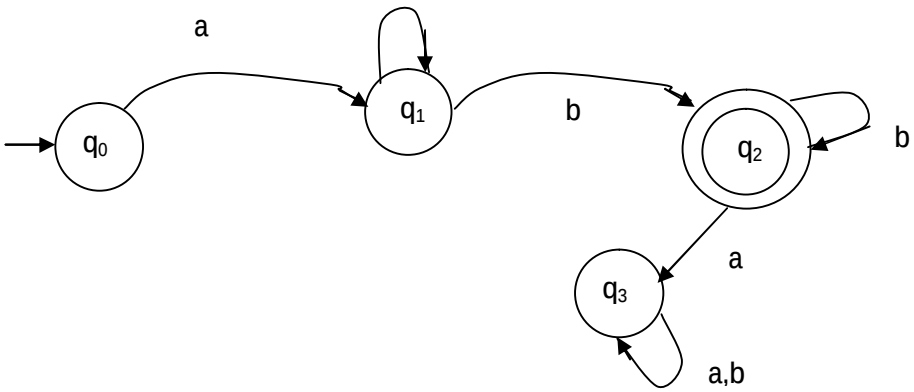
مثال) $L=a^*b^*$ و $\sum=\{ a ,b\}$



مثال) $L=a^*b^+$ و $\sum=\{a,b\}$



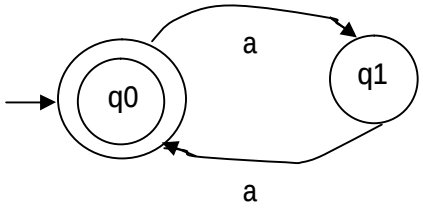
مثال) $L=a^+b^+$ و $\sum=\{a,b\}$



-برای زبان زیر یک Dfa طراحی کنید.

$L=(aa)^*$

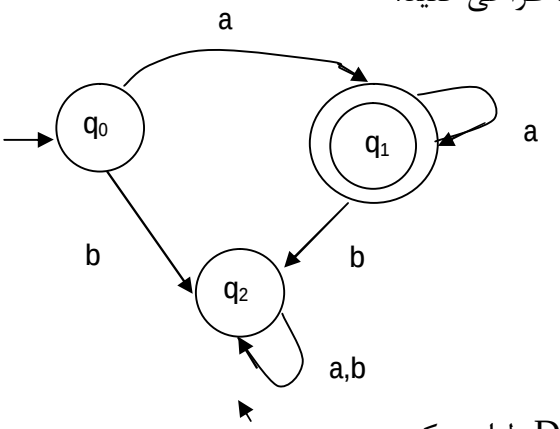
$\sum=\{a\}$



برای زبان زیر یک dfa طراحی کنید.

$L=a^+$

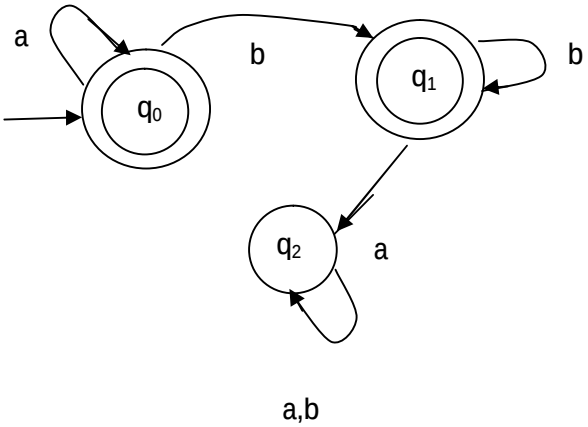
$\sum=\{a,b\}$



-برای زبان زیر یک Dfa طراحی کنید.

$L= a^* b^*$

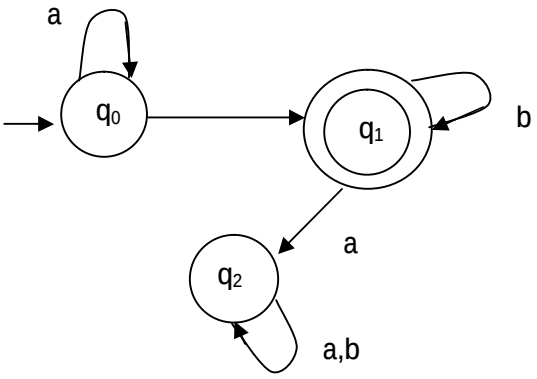
$\sum=\{a,b\}$



-DFA زبان زیر را طراحی کنید.

$L=a^* b^+$

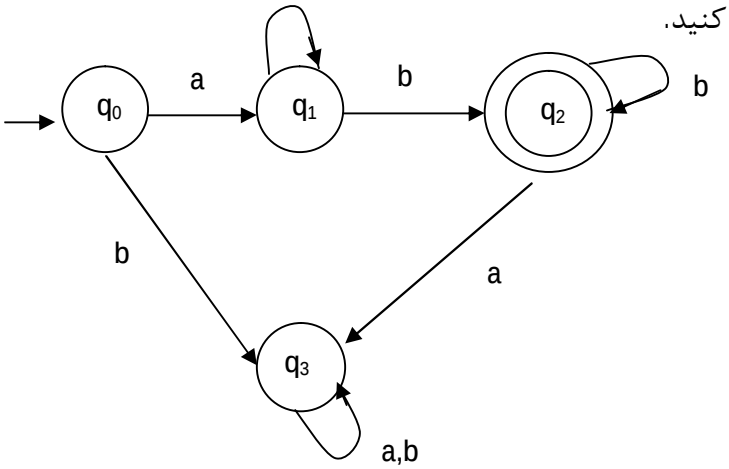
$\sum=\{a,b\}$



DFA- زبان زیر را طراحی کنید.

$L=a^+b^+$

$\sum=\{a,b\}$

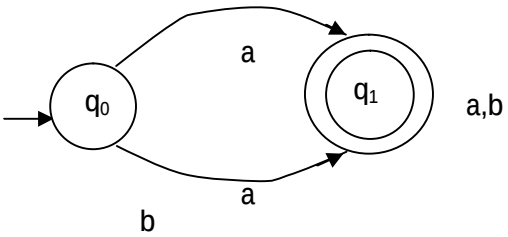


در DFA به ازای هر گره یا حالت باید به تعداد حروف الفبا یال خروجی داشته باشد.

DFA- زبان های زیر را بنویسید.

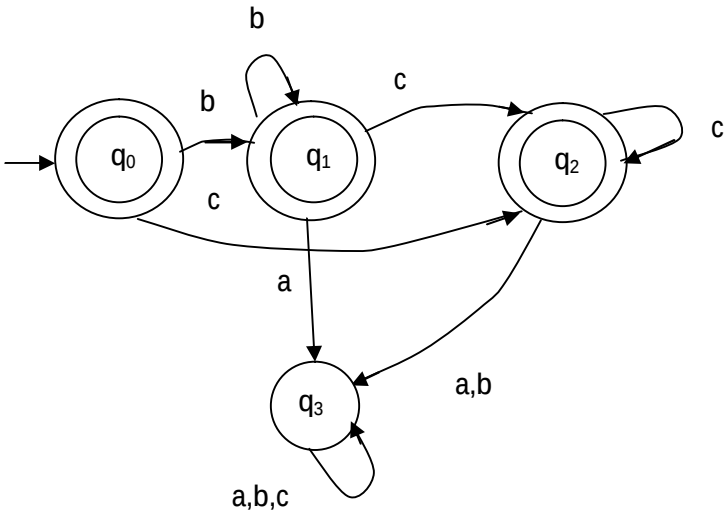
1) $L=a \cup b^+$

$\sum=\{a,b\}$



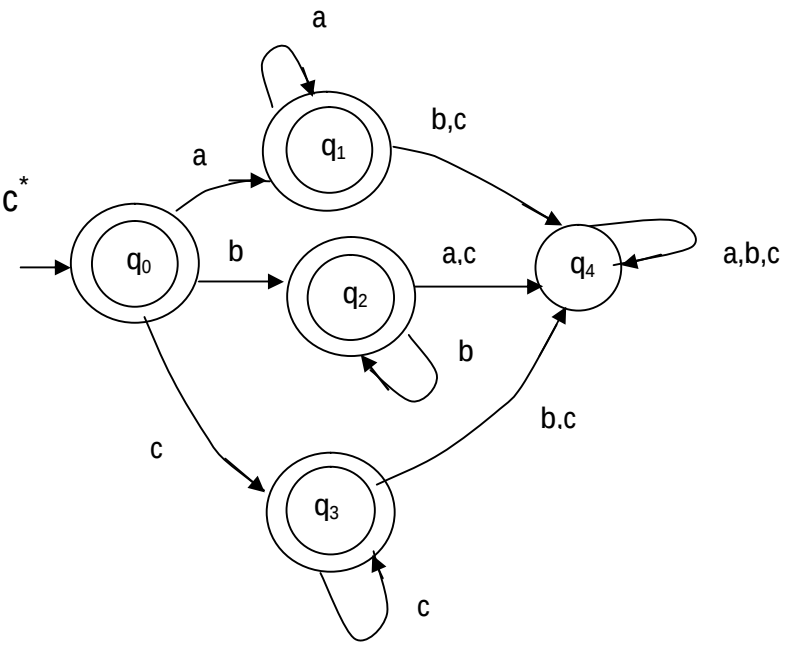
2) $L=a^*b^*c^*$

$\sum=\{a,b,c\}$



3)L=a* U b* U c*

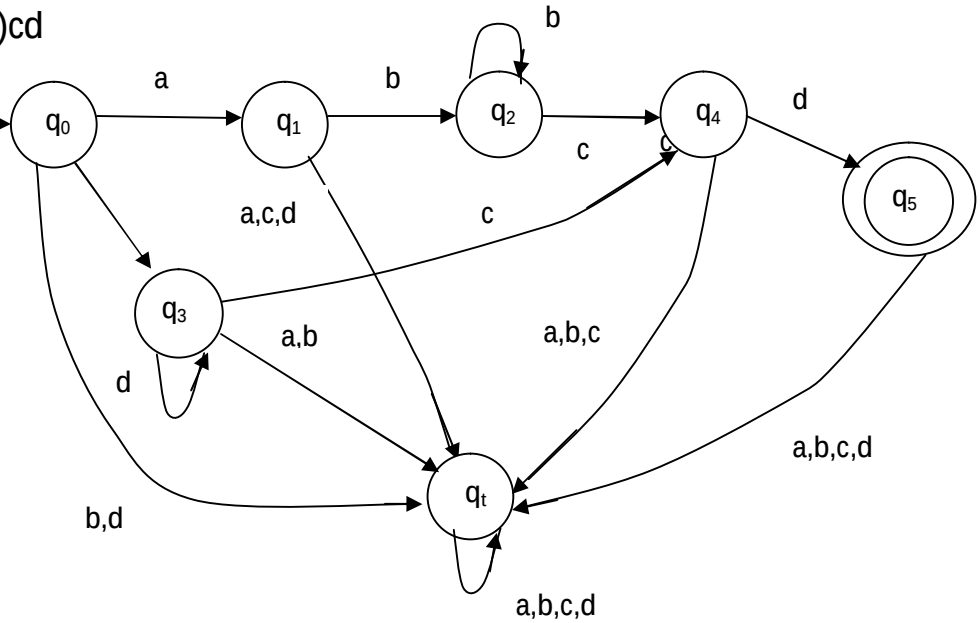
∑={a,b,c}



ü برای هر زبان با قاعده حداقل یک ماشین DFA وجود دارد که رشته های آن زبان را پذیرش می کند.

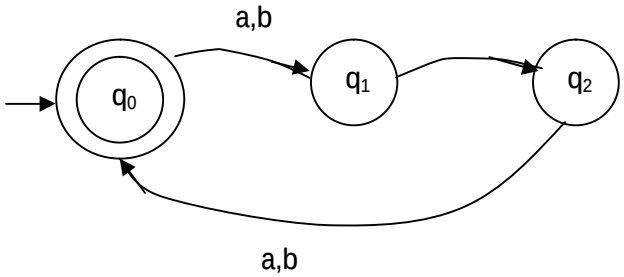
4)L=(ab+Ucd*)cd

∑={a,b,c,d}



5) L={w | w ∈ $\sum^* \ and \ \ lwl \ mod \ 3 \ = \ 0$ }

∑ = {a,b}

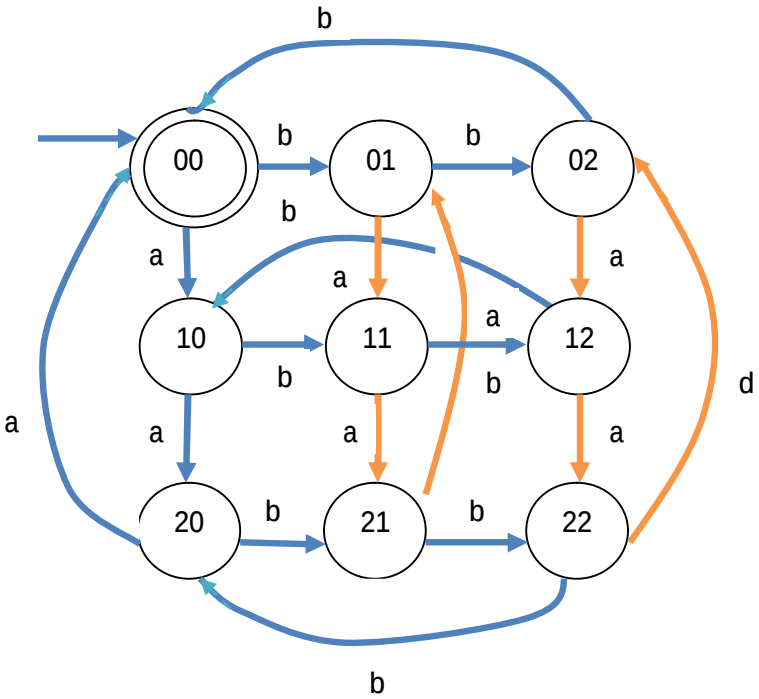


رشته های سه گانه ، (مضرب 3-6-9و...)

6) ∑={a,b}

L={w | Na $mod \ 3 \ = \ 0 \ and \ Nb \ mod \ 3 \ = \ 0$}

تعداد a,b ها باید مضرب سه باشد.



ماشین های متناهی غیر قطبی NFA : یک پنج تایی مرتب است که به صورت زیر تعریف می شود.

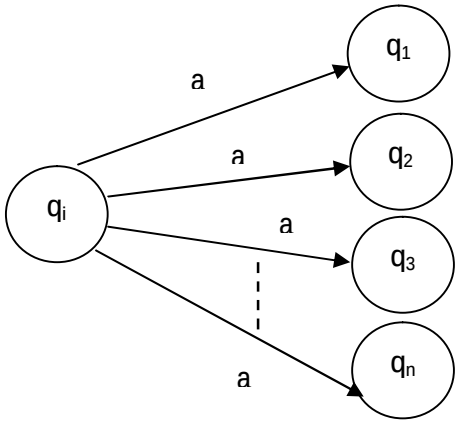
$M=(\sum , q_0 , Q , S , f)$

$\delta = Q * \sum \longrightarrow TQ$

مجموعه حالات: TQ

- در NFA با خواندن یک الفبا از ورودی الزاما فقط و فقط به یک حالت دیگر نمی رود یعنی در DFA الزاماً به ازای هر الفبا یک یال باید از هر گره خارج شود ولی در NFA الزامی به این کار نیست.

1) $\delta(q_i , a)$
2) $\delta (q_i , a)=\{ q_1,q_2,...,q_n\}$



هر DFA قطعا یک NFa است b ، هر NFA الزاماً یک DFA نمی باشد.

تعریف محاسبه موفق : محاسبه ای که پس از خواندن آخرین ترمینال ماشین به یکی از حالت های نهایی برسد.

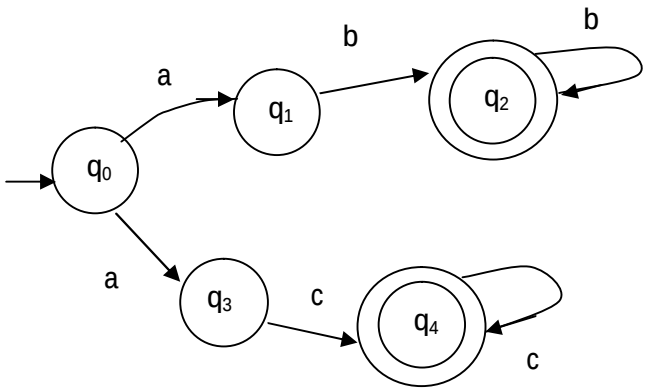
در DFA برای هر رشته فقط یک محاسبه وجود دارد که ممکن است موفق یا نا موفق باشد اما در NFA برای هر رشته ممکن است یک محاسبه موفق و یک محاسبه نا موفق وجود داشته باشد.

- در ماشین های NFA پیرامون عدم پذیرش رشته های نا معتبر بحث نمی شود و تنها در رابطه با نحوه پذیرش رشته های معتبر بحث می شود. یعنی در ماشین های NFA دیگر حالت تله نیاز نیست زیرا در مورد عدم پذیرش رشته نا معتبر بحث نمی شود.

-برای زبان زیر یک NFA طراحی نمایید.

$L = ab^{+} \cup ac^{+}$

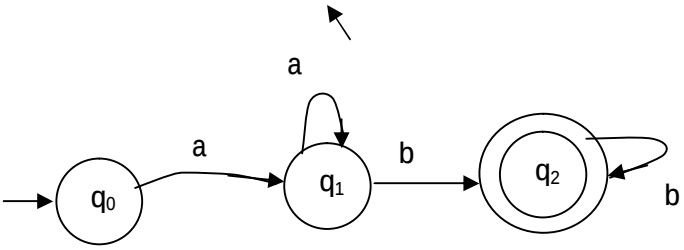
$\sum = \{a,b,c\}$



شرط پذیرش در NFA : یک رشته به یک NFA تعلق دارد اگر به ازای آن رشته حداقل یک محاسبه موفق وجود داشته باشد.

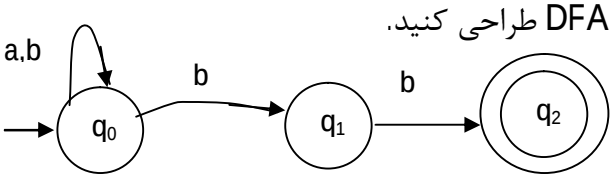
برای زبان زیر یک NFA طراحی کنید.

$L = a^{+} b^{+}$

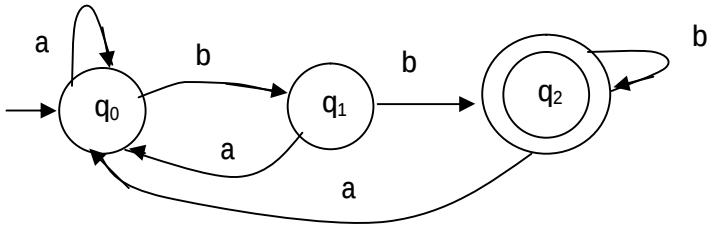


برای زبان $L= (a \cup b)^{*} bb$ یک NFA و DFA طراحی کنید.

NFA ⇨



DFA ⇨



ماشین متناهی غیر قطبی نوع لاند (λ -NFA): این نوع ماشین NFA است که می تواند بدون خواندن تعدادی از ورودی محتویات ثبات وضعیت خود را تغییر دهد.

$M = (\sum, q, Q, s, F)$

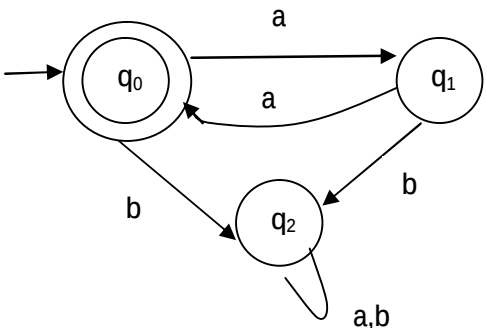
$\delta : Qx (\sum U \{ \lambda \}) \longrightarrow TQ$

$\delta (qi , \lambda) = \{q_1 , q_2 , \dots , q_n\}$

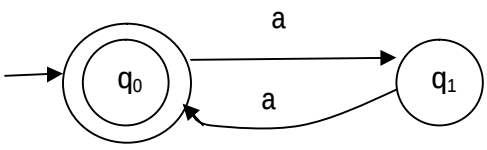
سوال؟

$L = (aa)^*$

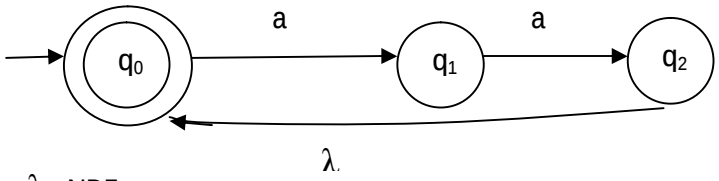
$\sum = \{a,b\}$



DFA



NFa



λ - NDF

برای زبان زیر یک NFA – λ طراحی کنید.

$L = a^+ b^+ d^+ U a^* c^+ b^+$