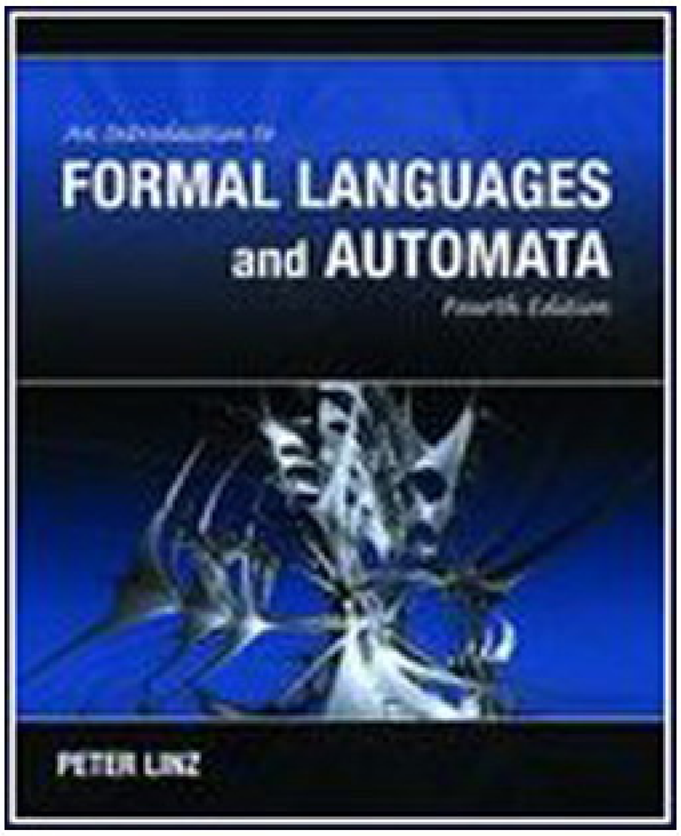
جزوه درس نظریه زبان ها و ماشین ها

استاد: دکتر علی نورالله ... دانشگاه آزاد قزوین

کتاب مرجع :

an Introduction to Formal languages and Automata



نظریه زبان ها و ماشین ها

فصل ۱ - مقدمه ای بر محاسبات :

مجموعه تمام زیرمجموعه های S

$$2^S = \{\phi, \{1\}, \{2\}, \{1,2\}\}$$

$S = \{1,2\}$

حاصلضرب دکارتی دو مجموعه (زوج مرتب از دو مجموعه)

$$S = S_1 \times S_2 = \{(x,y): \ x \in S_1 \ , y \in S_2\}$$

قوانین دمورگان

$$\overline{S_1 \cap S_2} = \overline{S_1} \cup \overline{S_2}$$

$$\overline{S_1 \cup S_2} = \overline{S_1} \cap \overline{S_2}$$

اگر دو مجموعه هیچ عضو مشترکی نداشته باشند → $S_1 \cap S_2 = \phi$

تابع: عناصر S_1 بصورت یکتا به عناصر S_2 ربط دارند: $f: S_1 \to S_2$ یعنی دامنه از S_1 برد از S_2 است.

رابطه: کلی تر از تابع است و یک عضو از S_1 می تواند به چند عضو S_2 ربط داشته باشد.

رابطه هم ارزی: بصورت $x \equiv y$ نشان می دهیم

خواص
1. بازتابی: $x \equiv x \quad \forall x$
2. تقارنی: if $x \equiv y$ then $y \equiv x$
3. ترایایی: if $x \equiv y$, $y \equiv z$ then $x \equiv z$

گراف: از مجموعه رئوس V و یال ها E تشکیل شده است.

درخت: نوع خاصی از گراف که چرخه ندارد و یک گره خاص بنام ریشه دارد.

روش های اثبات:
- استقرا
- برهان خلف

مفاهیم اساسی: زبان ها - گرامرها - آتاماتا (ماشین ها)

الفبا ← یک مجموعهٔ غیرتهی از نشانه هاست. مثلاً $\Sigma=\{a,b\}$

رشته w ← از الفبا تشکیل می شود: مثل aba و babb

- طول رشته را با $|w|$ نشان می دهند
- معکوس رشته را با w^R نشان می دهند
- رشته تهی را با λ نشان می دهند
- $\forall w: \lambda w = w\lambda = w \longrightarrow |\lambda| = 0$

اتصال ← یعنی دو رشته رو بهم بچسبونیم

زیررشته ← بخشی از حروف پشت سرهم یک رشته.

تکرار رشته ← تکرار n بار رشته w را بصورت w^n نشان می دهند. پس $w^0=\lambda$ می شود.

Σ^* ← مجموعه تمامی رشته هایی که با اتصال صفر یا بیشتر از حروف Σ بدست می آید. (شامل λ هم هست).

Σ^+ ← شامل λ نیست.

Σ متناهی است. اما Σ^+ و Σ^* نامتناهی هستند.

زبان ← زیرمجموعه ای از Σ^* است. و به یک رشته در یک زبان، یک جمله می گوئیم. زبان را با L نشان میدهیم

$\Sigma=\{a,b\}$ $\quad L=\{a^nb^n : n\geq 0\} \longrightarrow$ پس aabb یا aaabbb در این زبان هستند. اما abb نیست.

زبان ها متمم دارند $\bar{L}=\Sigma^*-L$ وهمچنین عکس دارند: $L^R=\{w^R : w\in L\}$

- اتصال دو زبان بصورت $L_1L_2=\{xy : x\in L_1, y\in L_2\}$
- زبان ها مجموعه هستند، پس روی آنها اجتماع و اشتراک و تفاضل قابل تعریف است.
- L^n به معنای اتصال L به خودش، n بار است.

گرامرها: گرامر G بصورت زیر تعریف می‌شود:

$G = (V, T, S, P)$

- V: Var متغیر
- T: Terminal نشانه
- S: Start $\in V$ متغیر شروع
- P: قوانین

- T, V غیرتهی و مجزا هستند.
- قوانین قلب گرامر هستند. مثل $(x \to y)$

در اینجا $x \in (V \cup T)^{+}$ و $y \in (V \cup T)^{*}$ است. اگر داشته باشیم $w = uxv$ و $x \to y$ آنگاه

$z = uyv$ این بدست می‌آید:

پس می‌شه گفت که w ، z را می‌دهد:

$w \Rightarrow z$

زبان: اگر گرامر $G(V,T,S,P)$ را داشته باشیم ، آنگاه مجموعه $L(G)$ زبان تولیدشده توسط G است.

$$L(G) = \{ w \in T^{*} : S \stackrel{*}{\Rightarrow} w \}$$

سازی مشتق می‌شود.

مثال) $G(\{S\}, \{a,b\}, S, P)$

$P: S \to aSb \mid \lambda$

$S \Rightarrow aSb \Rightarrow aaSbb \Rightarrow aabb$ و

پس می‌توان گفت $S \stackrel{*}{\Rightarrow} aabb$

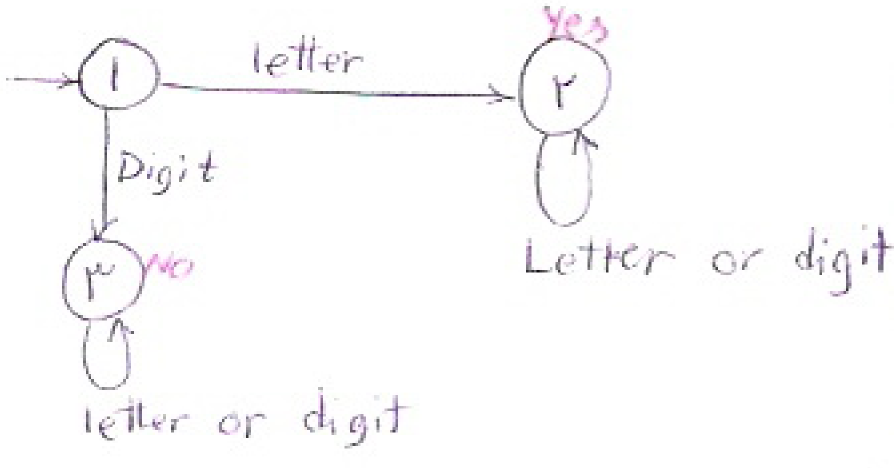
در اینجا گرامر ، زبان را نشان می‌دهد: $L(G) = \{a^n b^n : n \geq 0\}$

اما همیشه اینجوری نیست.

⊛ برای اینکه ثابت کنیم یک گرامر توسط یک زبان تولید می‌شود باید ثابت کنیم
① $w \in L$ برای تمام رشته‌ها می‌توانند توسط گرامر تولید شوند
② هر جمله‌ای که توسط گرامر تولید می‌شه در L هست

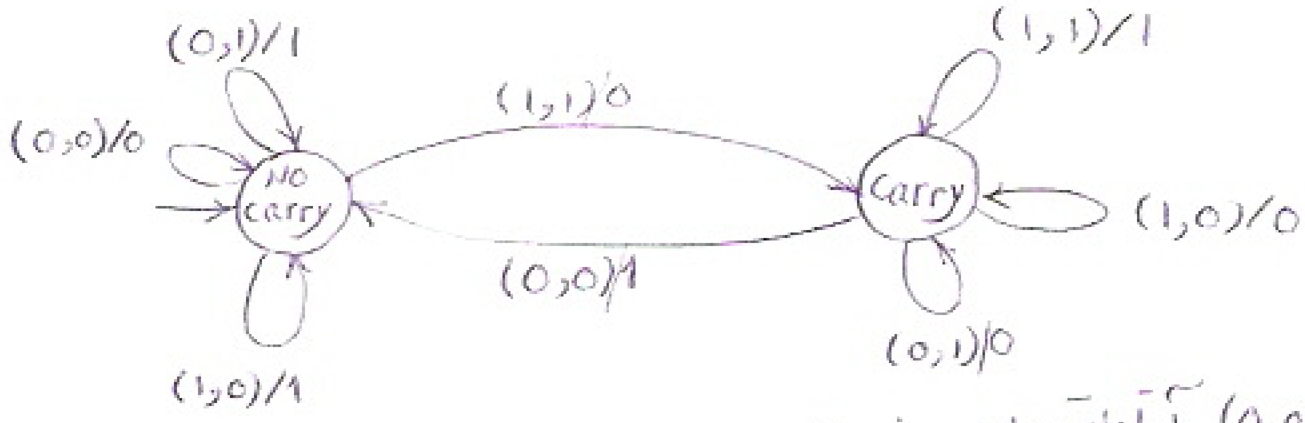
آتاماتا ← یک مدل انتزاعی از کامپیوتر است. که ورودی می‌گیرد . هر ورودی باعث تغییر وضعیت کنونی آتاماتون می‌شود - آتاماتون در هر وضعیت حروف خاصی از ورودی را می‌خواند و تغییر وضعیت می‌دهد که حرکت گویند.

مثلاً: متغیرهای پاسکال را بصورت آتاماتا نمایش دهید



این می‌گه که اگر متغیری با حرف شروع شود به وضعیت ۲ می‌رود و بعدش اگر حرف یا رقم باشد در هر دو حالت پذیرفته است . اما اگر حرف اولی رقم باشد به وضعیت ۳ می‌رود و بعدش هر چی بیاد هم همون وضع ۳ است که رد شده است.

رسم آتاماتون: بصورت گراف نشان می‌کنیم - البته یال های آن را با $(a_i , b_i)/d_i$ علامت گذاری می‌کنیم
(ورودی ، خروجی)



همانطور که می بینید دریافت زوج (0,0) آتاماتون را به وضعیت carry می برد

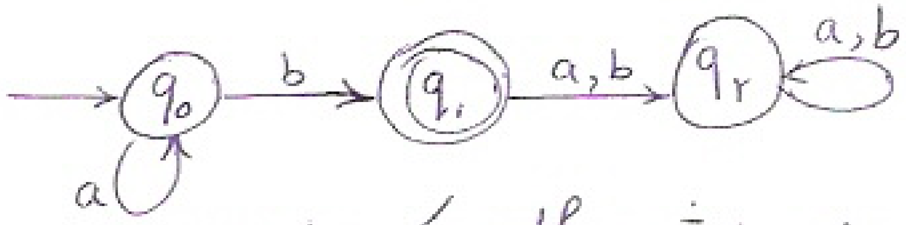
فصل ۲ : آتاماتای متناهی

FA ← آتاماتای متناهی - (قابلیت ذخیره سازی محدود دارد)

dfa ← اول در q_0 است و با هر ورودی، تغییر وضعیت می دهد. اگر وقتی تمام ورودی ها تمام می شود، در وضعیت نهایی ◎ باشد، می گوئیم رشته ورودی را پذیرفته

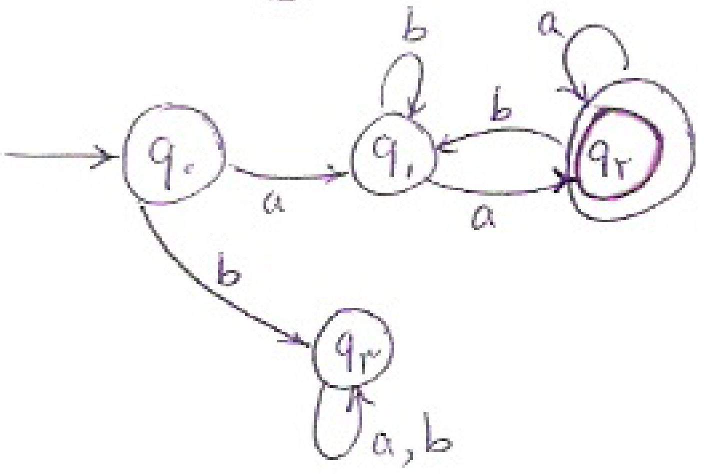
زبان dfa : یعنی مجموعه تمامی رشته هایی که dfa می پذیرد. مثلاً زبان ماشین زیر بصورت زیر است:

$L(M)=\{a^n b : n \geq 0\}$



زبان منظم : زبانی که بتوان برایش dfa رسم کرد منظم است.

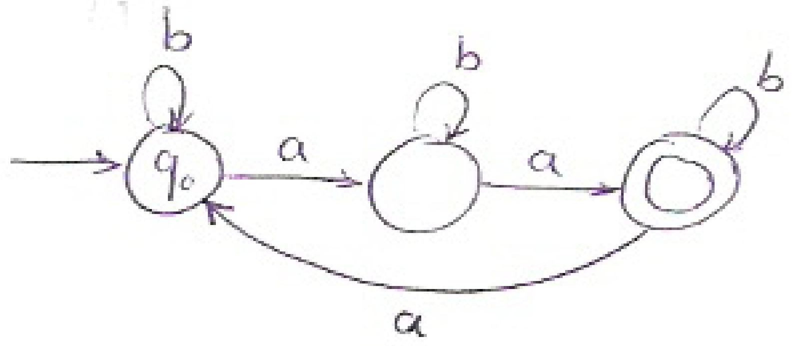
مثال) نشان دهید که زبان $L=\{awa : w\in\{a,b\}^*\}$ منظم است: برای اینکار باید بتوانیم یک dfa برای این رسم کنیم



این یعنی شروع و پایان با a باشد و بین آنها w باشد که w می تواند از حروف a,b تشکیل شود و چون * ستاره دارد می تواند هیچی هم باشد.

مثال) برای زبان زیر روی $\Sigma=\{a,b\}$ یک dfa بیابید:

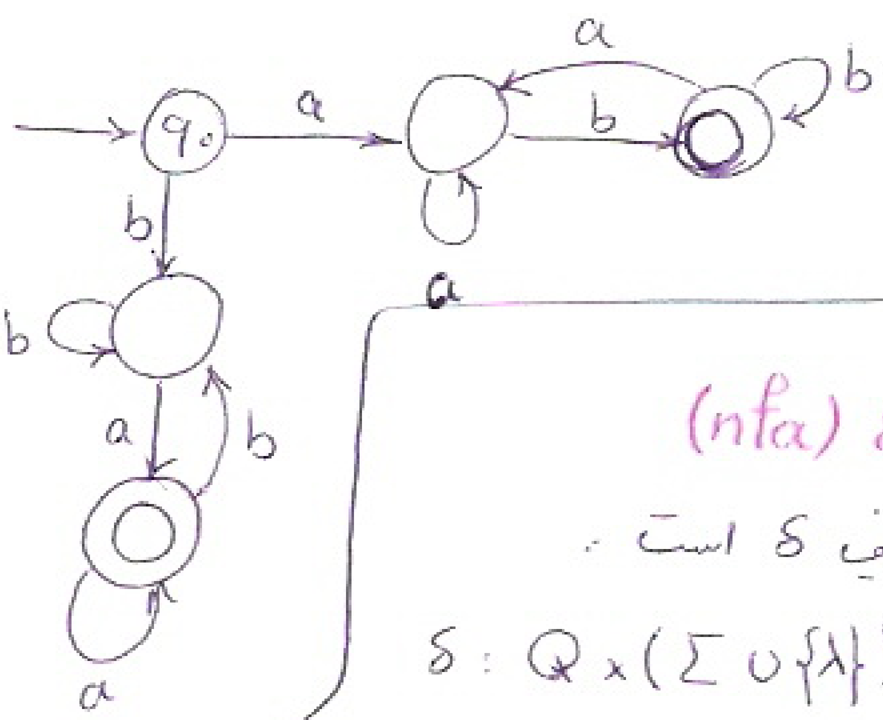
$L=\{n_a(w) \bmod 3 \geq 1\}$



حل) یعنی تعداد a در جمله w را اگر بر ۳ تقسیم کنیم، باقیمانده بیش از 1 داشته باشد. باقیمانده های تقسیم بر ۳ عبارتند از 0 و 1 و 2 پس فقط 2 می ماند.

مثال) یک dfa رسم کنید که رشته هایی را بپذیرد که اولین حرفش با آخرین حرفش فرق داشته باشد:

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



پذیرنده ناشناختی نامعین: (nfa)

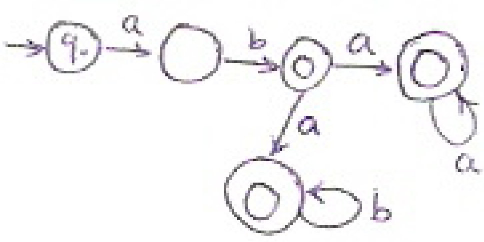
تفاوت آن با dfa در تعریف δ است.

$$\delta : Q\times(\Sigma\cup\{\lambda\}) \longrightarrow 2^{Q}$$

درحالیکه در dfa داشتیم: $Q\times\Sigma\rightarrow Q$

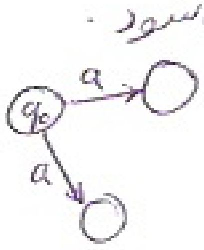
مثال) یک آتاماتون nfa برای زبان زیر بکشید:

$\{abab^n : n\geq 0\}\cup\{aba^n : n\geq 0\}$



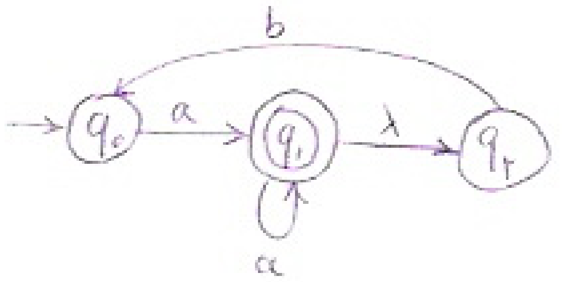
NFA

- در حالت nfa: λ پذیرفته می شود و می تواند باعث تغییر وضعیت شود.
- برای یک ورودی، می توان به بیش از یک وضعیت رفت ←



- می تواند $\delta(q_0,a)$ تهی باشد. یعنی با a به هیچ کجا نرفت!

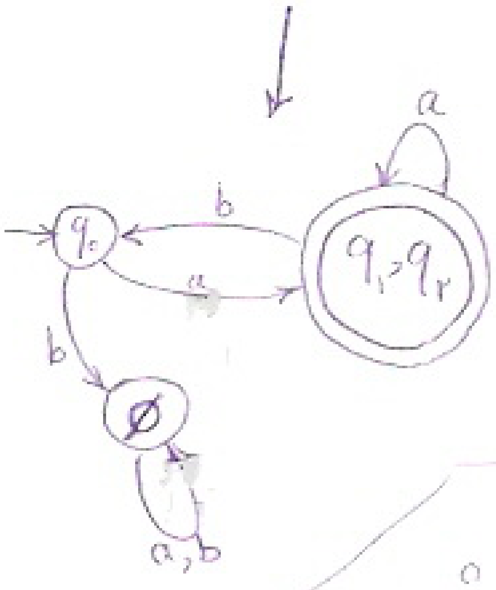
تساوی dfa و nfa: دو آتاماتون مساوی هستند، اگر هر دو یک زبان را بپذیرند.



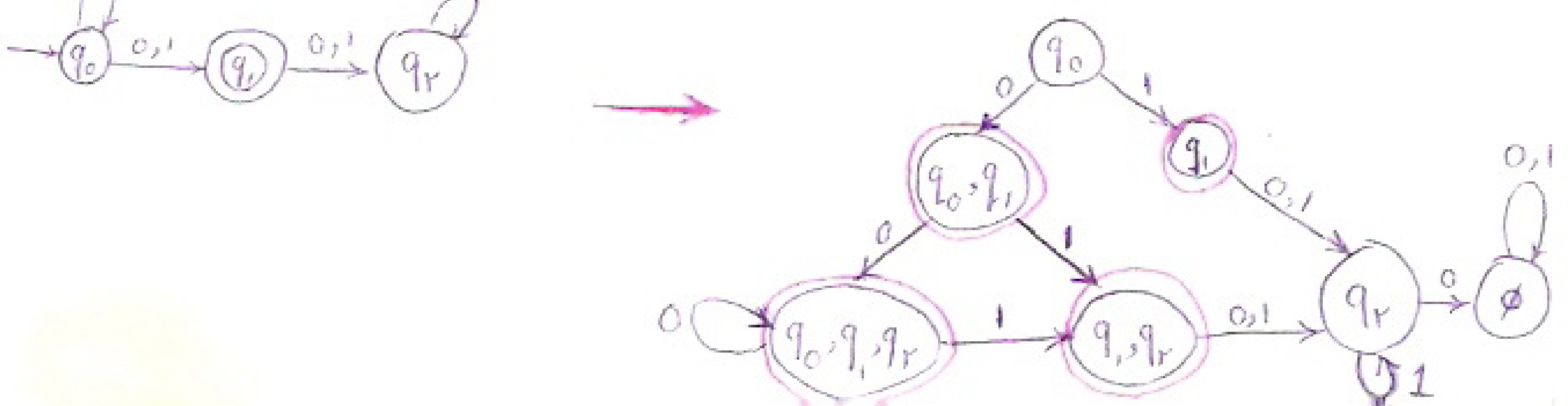
تبدیل nfa به dfa: در آتاماتون روبرو که nfa است دقت کنید.

— در q_0 با ورودی a به q_1 رفته و با λ به q_2 می رسیم. پس در dfa معادلش با a از q_0 به q_1,q_2 می رسیم.

— در q_0 با b حرکتی انجام نمی شود. پس در dfa معادل یک وضعیت تهی می کشیم که با b به آن می رویم و غیرنهایی (تله) است.

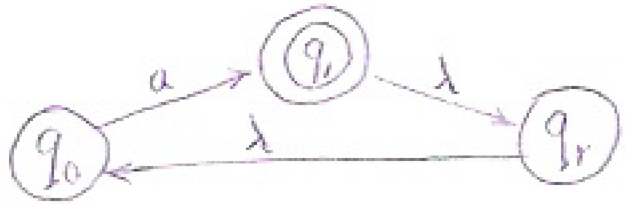


مثال) nfa زیر را به dfa تبدیل کنید:



* چون زبان هر nfa توسط یک dfa پذیرفته می شود ⟸ زبان nfa منظم است.

تمرین (نکته) در شکل زیر $\delta^*(q_0, a)$ و $\delta^*(q_1, \lambda)$ را بیابید:



$$\delta^*(q_0, a) = \{q_1, q_2, q_0\}$$

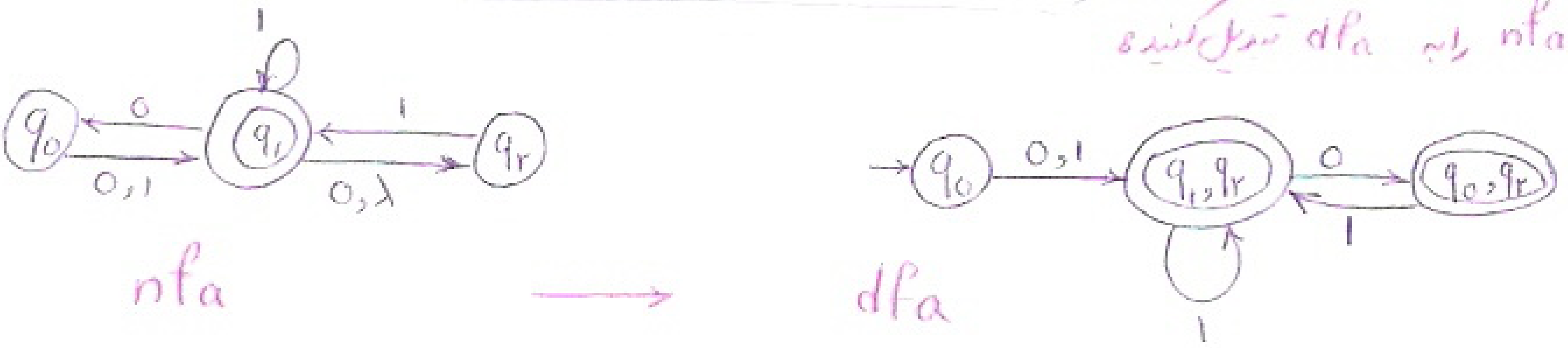
$$\delta^*(q_1, \lambda) = \{q_1, q_2, q_0\}$$

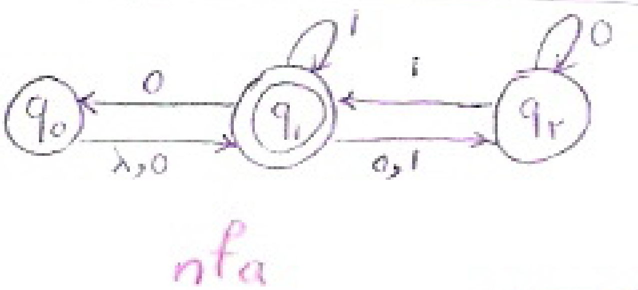
حالت شروع

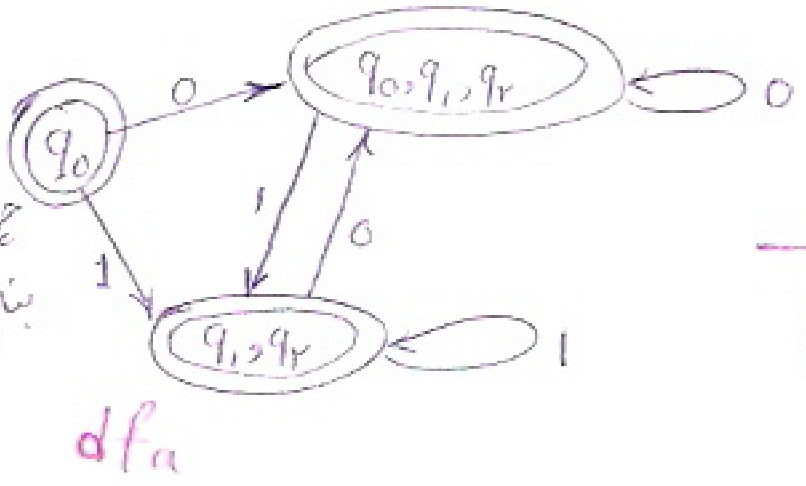
ورودی

q_1 با لاندا به خودش هم می تواند برود.

تمرین) nfa را به dfa تبدیل کنید

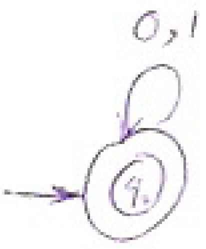






چون q_0 با λ به حالت نهایی می رود بنابراین در dfa حالت q_0 نهایی است

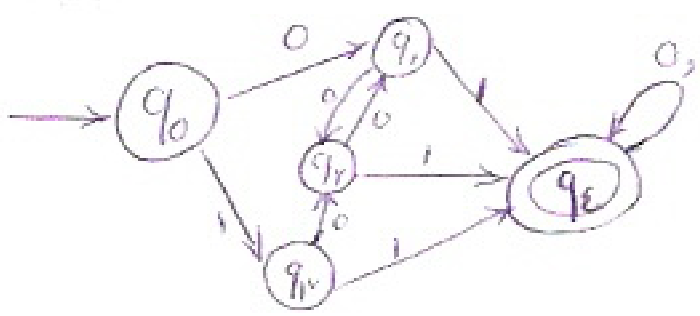
نهایتاً

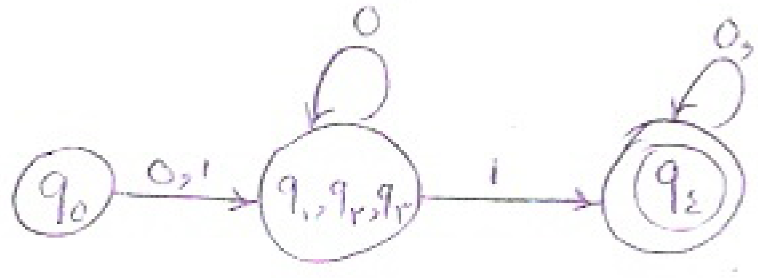


کاهش تعداد وضعیت های FA

یک dfa یک زبان خاص را می پذیرد - اما یک زبان می تواند توسط چندین آتاماتون پذیرفته شود - گاهی آتاماتونی که به ما می دهند می تواند با وضعیت های کمتر، همان زبان را پذیرش کند. با انجام این اصول زیر می توان یک dfa را کاهش وضعیت داد.

مثال

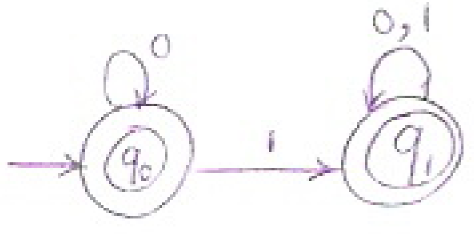




اصول کاهش وضعیت‌ها:

① اگر وضعیتی در dfa هست که اصلاً نمی‌شه واردش شد، حذفش کن (چون اصلاً در پروسه نیست)

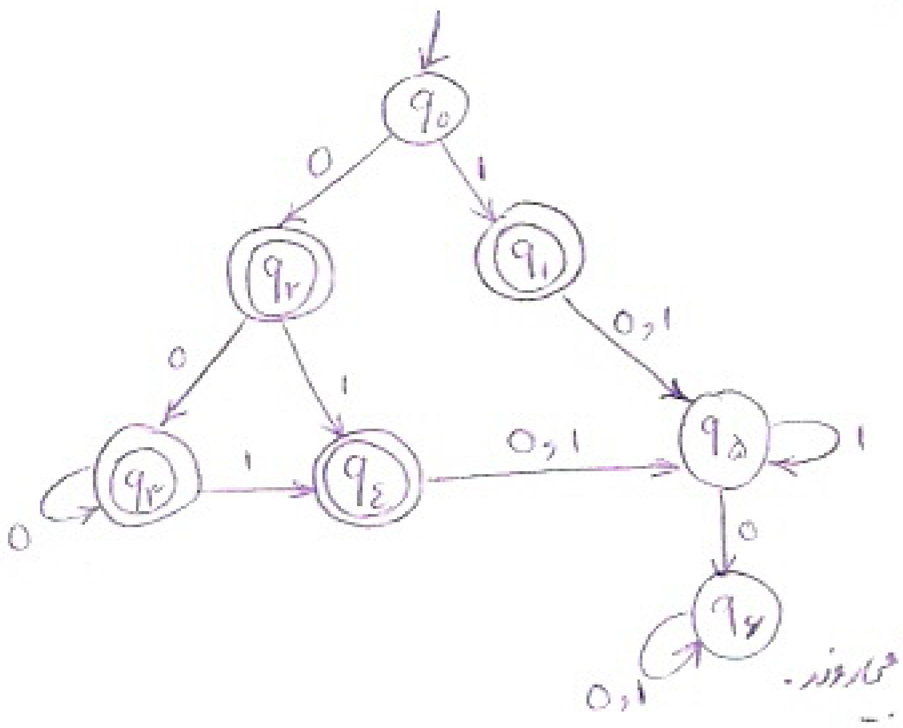
② دو حالت q_0 و q_1 را در نظر بگیرید - اگر ورودی ای به q_0 بدهیم که اگر به q_1 هم بدهیم به نهایی برود و نیز ورودی ای به q_0 بدهیم که اگر همان را به q_1 بدهیم به غیر نهایی بروند آنگاه این دو وضعیت شبیه هم عمل می‌کنند و ادغام پذیرند.



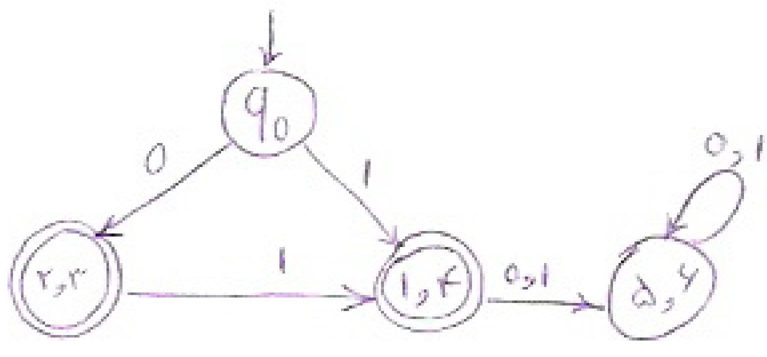
③ اگر دو حالت، یکی نهایی باشد و دیگری غیر نهایی، هرگز ادغام نمی‌شوند.

④ اگر حالات ادغام ناپذیر را پیدا کنید، بقیه حالات ادغام پذیرند.

مثال: dfa زیر را خلاصه کنید:



⟹



q_3 و q_2 با هم ترکیب شدند. چون با 0 واردشان می‌شویم و با 0 و 1 به نهایی می‌روند.

q_4 و q_1 با هم ترکیب شدند. چون با 1 واردشان می‌شویم و با 0 و 1 به یک نقطه می‌روند.

q_6 و q_5 با هم ترکیب شدند. چون هر دو با 0 و 1 فقط به غیر نهایی می‌روند.

فصل سوم (زبان‌ها و گرامرهای منظم)

⊛ زبان منظم می‌تواند توسط یک ماشین nfa یا dfa توصیف شود.

عبارات منظم: شامل حروف زبان مثل $\{a,b\}$ و همچنین عملگرهای + و . و * است.

تقدم عملگرها
*
.
+

+ ⟵ عمل اجتماع را نشان می‌دهد (or)

. ⟵ نقطه برای اتصال است (and). نیازی به نوشتن نقطه نیست.

* ⟵ ستاره ستاره‌وار (یعنی می‌تواند نباشد، باشد، یا چندین بار باشد)

$\emptyset$ و λ و همچنین حروف الفبا مثل $\{a,b\}\in\Sigma$ عبارات منظم ابتدایی هستند.

- اگر r_1 و r_2 عبارات منظم باشند، r_1+r_2 و $r_1.r_2$ و r_1^* و (r_1) عبارات منظم هستند.

- اگر رشته را بتوان با $\emptyset$ و λ و Σ و قوانین فوق ↑ تولید کرد، اونم منظم میشه.

مثال) عبارت $(c+\emptyset)(a+b+c)^*$ منظم است. اما عبارت $(a+b+)$ نامنظم است.

زبان منظم) اگر r عبارت منظم باشد $L(r)$ بصورت زیر تعریف می‌شه:

شامل $\emptyset$، $\{\lambda\}$، تمامی الفبا مثل $\{a\}$ عبارت منظم هستند و

$$L(r_1+r_2)=L(r_1)+L(r_2)$$

$$L(r_1r_2)=L(r_1).L(r_2)$$

$$L(r_1^*)=(L(r_1))^*$$

مثال) $r=(a+b)^*(a+bb)$

این پرانتز فقط یکبار اجرا می‌شود. و ما این حق را داریم که a یا bb برداریم.

یکی از این دو می‌آید چون بین آنها + است. همچنین چون * دارد می‌تواند اصلاً نیاید یا چندین بار اجرایش کنیم. مثلاً بار اول a برداریم. بار دوم b برداریم و بار سوم a برداریم نتیجه بصورت aba می‌شود.

در نهایت رشته ما حتماً به a یا bb ختم می‌شود. اما می‌تواند با چیزهایی که از پرانتز ستاره‌دار اول می‌سازیم شروع شود.

پس اگر بخواهیم زبان آن را بصورت مجموعه‌ای نشان دهیم بصورت زیر می‌شه

$L(r)=\{a, bb, aa, abb, ba, bbb, \ldots\}$

مثال) برای $\Sigma=\{0,1\}$ عبارت منظمی بنویسید که دارای یک جفت صفر حتماً باشد:

$$\underbrace{(0+1)^*}_{\text{قبلش هرچیزی می‌تواند باشد}}\ \underbrace{00}_{\text{حداقل این جفت صفر هست.}}\ \underbrace{(0+1)^*}_{\text{بعدش هم هرچیزی می‌تواند باشد.}}$$

کوتاه‌ترین جمله این عبارت بصورت 00 است.

مثال) عبارت منظمی که دوتا صفر هرگز پشت سرهم نیایند.

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

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

مثال) برای مجموعه زیر عبارت منظم بنویسید:

$\{a^n b^m : m+n = \text{زوج}\}$

$m+n$ زوج ← یعنی یا هردو زوج یا هردو فرد.

$$\underbrace{(aa)^*}\ \underbrace{(ab+\lambda)}_{\text{برای فرد کردن } n \text{ و } m}\ \underbrace{(bb)^*}$$

برای تولید m زوج و n زوج

مثال) برای زبان زیر یک عبارت منظم بنویسید:

$L=\{a^n b^m : n\geq 4 \ ;\ m\leq 3\}$

$aaaaa^*(\lambda+b+bb+bbb)$

نکته) عبارت ϕ بصورت زیر است:

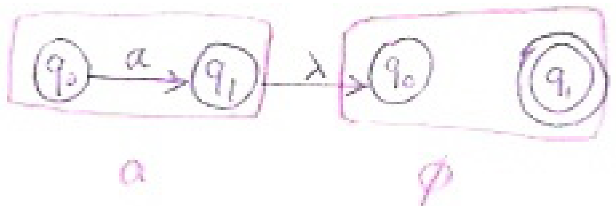
ϕ

توجه کنید که هر چیزی که * داشته باشد می‌تواند λ شود پس $\phi^*=\lambda$

عبارت a بصورت زیر می‌شود:

$\{a\}$

عبارت $a\phi$ به صورت زیر می‌باشد:



پس $a\phi$ به فاینال نمی‌رسد و $a\phi=\phi$

مثال) برای زبان زیر عبارت منظم بنویسید:

$L(r)=\{ab^n w \quad w\in\{a,b\}^+ \ ,\ n\geq 3\}$

$r=abbbb^*(a+b)(a+b)^*$

مثال) $\Sigma=\{a,b,c\}$ تمامی رشته هایی که حداکثر ۳ تا a دارند

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

مثال) $\Sigma=\{0,1\}$

تمام رشته هایی که شامل تعداد زوج صفر باشند:

$(1+01^*0)^*$

تمامی رشته هایی که حداقل ۲ رشته 00 دارند:

$(0+1)^*00(0+1)^*00(0+1)^*$

رشته هایی که شامل 101 نباشند:

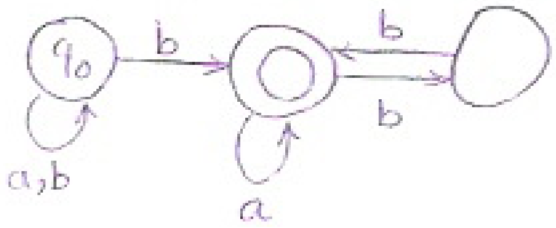
$(1+1000^*)^*(\lambda+10)$

ارتباط زبان منظم و عبارت منظم : هر زبان منظم ، عبارت منظم دارد و برعکس.

هر عبارت منظم توسط یک dfa پذیرفته می شود.

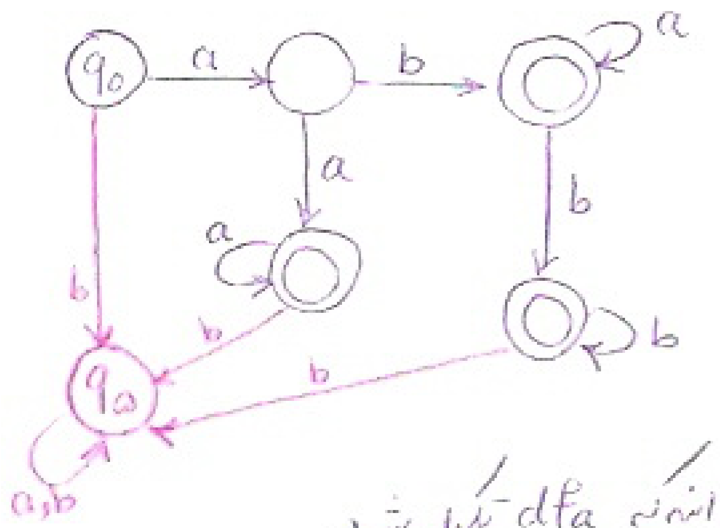
مثال) یک nfa بکشید که زبان زیر را بپذیرد:

$L = ((a+b)^* b (a+bb)^*)$



مثال) dfa بکشید:

$L = (aa^+ + aba^*b^*)$

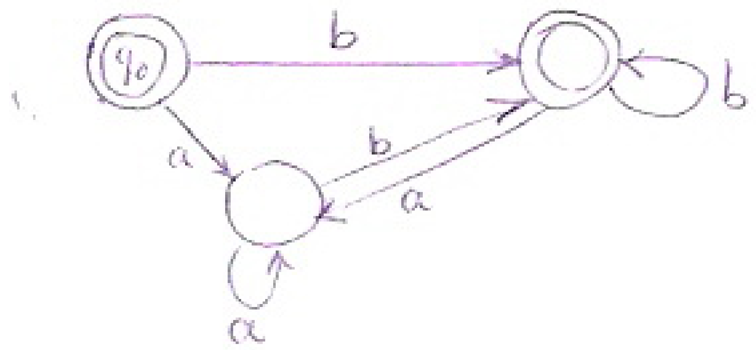


توجه) $q_\emptyset$ برای اینکه dfa تکمیل شود وگرنه در nfa لازم نبود رسم شود.

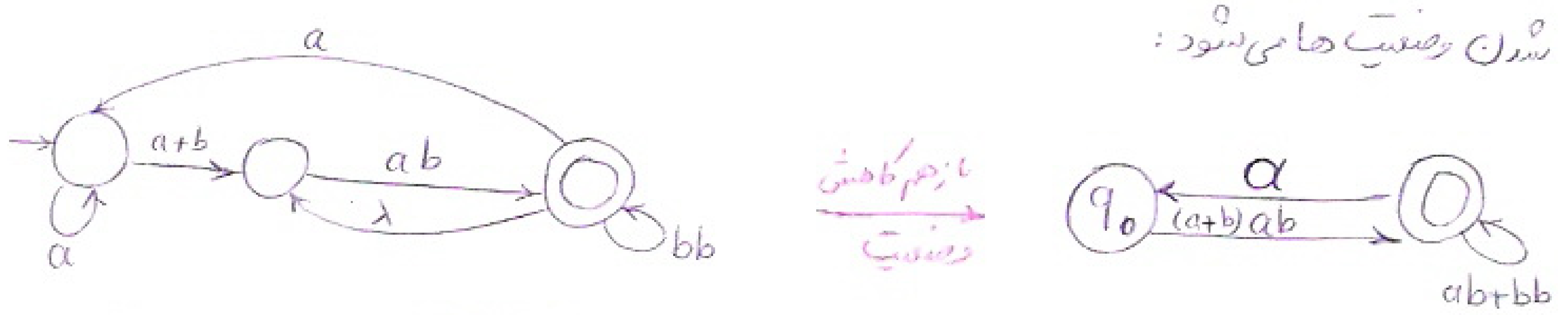
مثال) dfa رسم کنید

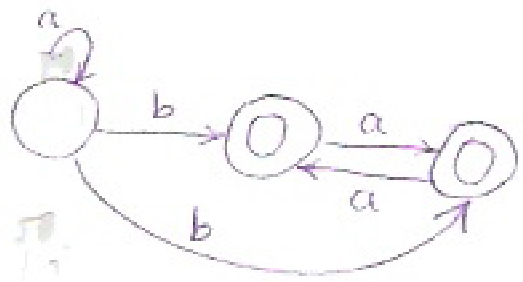
$L = ((aa^*)^*b)^*$

این زبان با $(a^*b)^*$ معادل است.

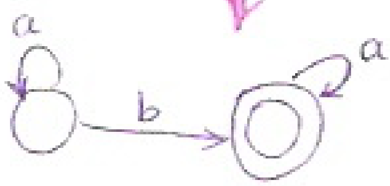


گراف تغییر وضعیت عام: گرافی که روی یال هایش ، به جای یک حرف ، یه عبارت نوشته شود. اینطور باعث کم شدن وضعیت ها می شود:



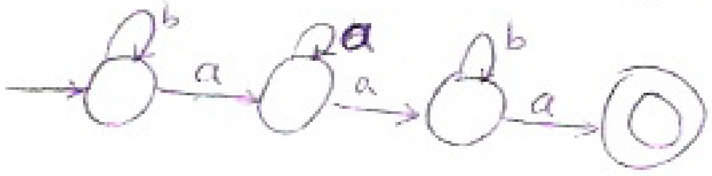


خلاصه کردیم



a^*ba^*

مثال) برای آتاماتون زیر عبارت منظم بنویسید:



$b^*aa^*ab^*a$

11

ص ۲۸ عبارت منظم بنویسید:

سوال ۱۵) الف) $(0+1)^* 01$ ← عباراتی که به 01 ختم شوند.

ب) $\lambda + 0 + 1 + (0+1)^*00 + (0+1)^* 10 + (0+1)^* 11$ ← رشته هایی که به 01 ختم نشوند

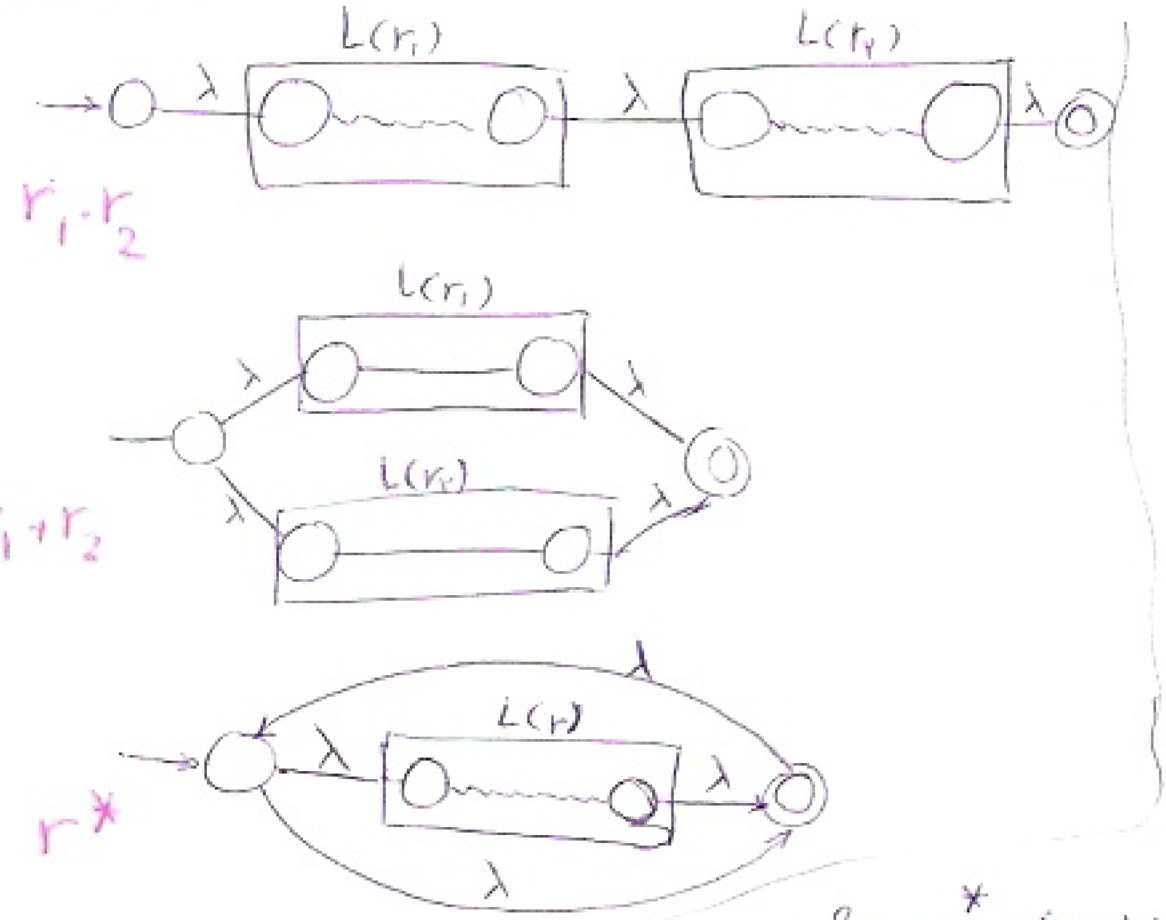
سوال ۱۶) بخش c) $b^*a\,(b^*ab^*ab^*ab^*ab^*ab^*)$ ← این باقیمانده‌اش 1 است.

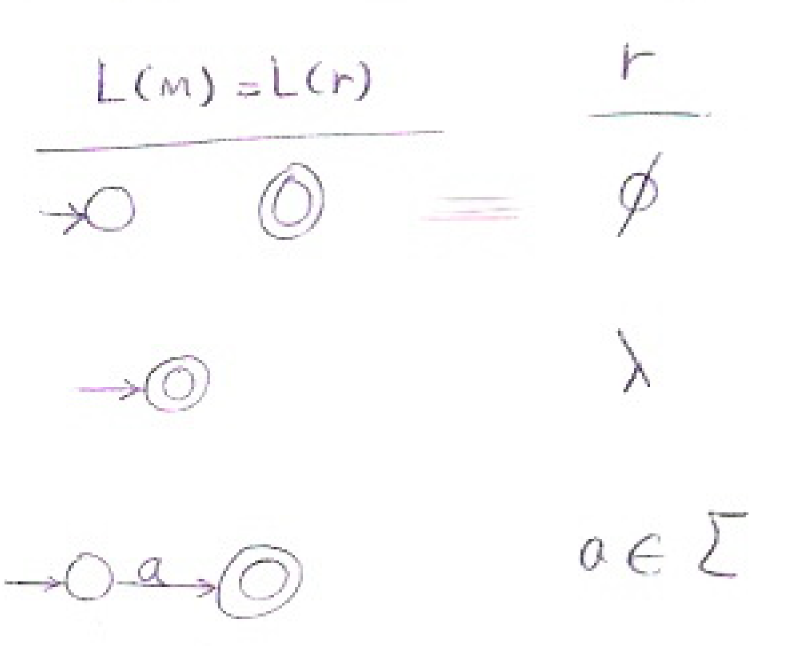
با 2 و 3 و 4 جمع کنید جواب مورد نظر می‌شه

مثال: عبارت منظم برای زبان روبرو: $L=\{w: n_a(w)$ و $n_b(w)$ هر دو زوج$\}$

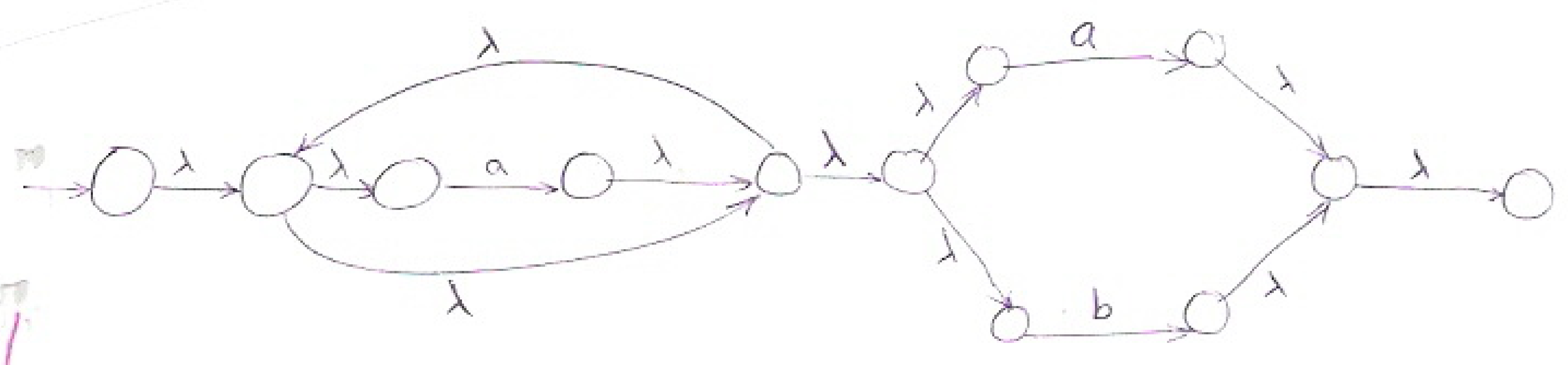
$$\left(aa+bb+(ab+ba)(aa+bb)^*(ab+ba)\right)^*$$

ارتباط بین عبارات منظم و ماشین‌های متناهی

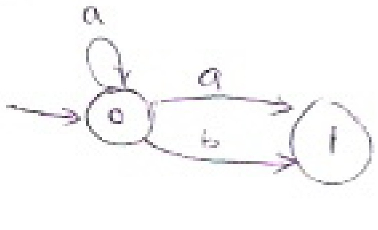




مثال) ماشین NDFA طراحی کنید که زبان آن $a^*.(a+b)$ باشد.



خلاصه شده این ماشین



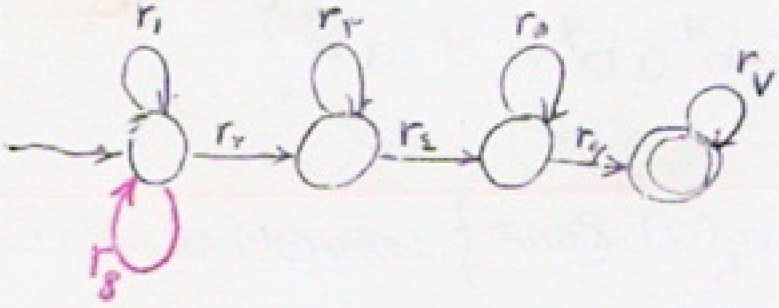
عبارت منظم را از ورودی گرفتیم و ماشین معادل آن را تولید کردیم.

مثال)

حالا می خواهیم از روی آتاماتون، عبارت منظم بنویسیم:

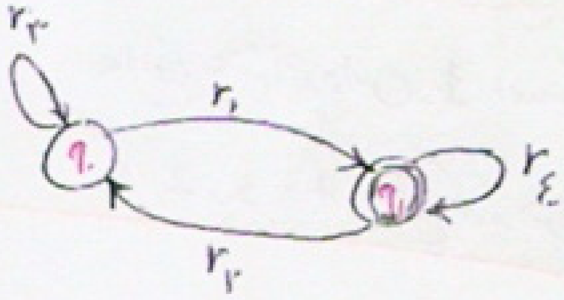
تمامی مسیرهایی که ما را از حالت اولیه به حالت نهایی می رساند را نوشته و با هم OR می کنیم.

* ستاره زمانی استفاده می شود که از یک نقطه به خودش می رسیم (حلقه)

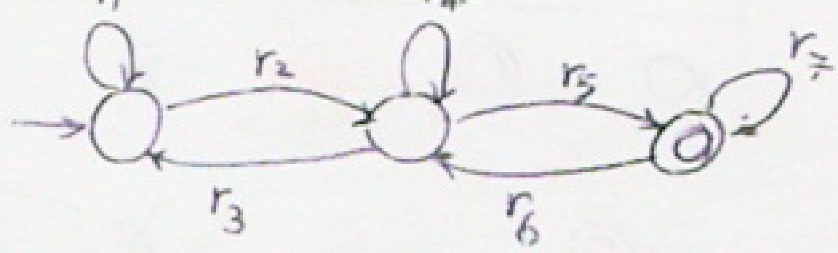


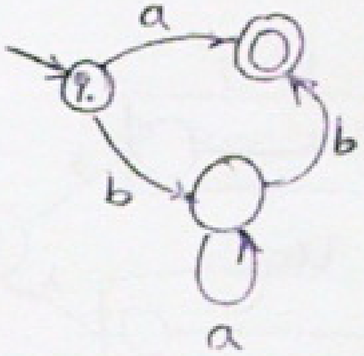
r_8 بدون: $r_1^* . r_2 . r_3^* . r_4 . r_5^* . r_6 . r_7^*$

r_8 با: $(r_1+r_8)^* . r_2 . r_3^* . r_4 . r_5^* . r_6 . r_7^*$



$(r_3^* + r_1 r_4^* r_2)^* . r_1 r_4^*$

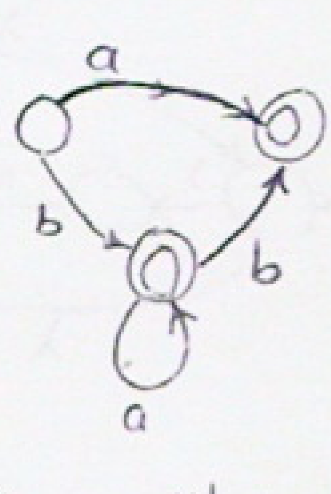




$(a + ba^* b)$

$(r_1^* + r_2 (r_4 + r_5 r_7^* r_6)^* r_3)^* r_2 (r_4 + r_5 . r_7^* r_6)^* . r_5 (r_7)^*$

وقتی دو تا حالت نهایی داریم، یکبار فرض می کنیم اولی نهایی است و یکبار فرض می کنیم دیگری نهایی است. دو حالت را جداگانه نوشته و با هم جمع می کنیم.



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

هر دو حالت درست است:

خلاصه شده

$(a + ba^* (b + \lambda))$

دومی، خلاصه شدهٔ همون بالایی است

13

$(a+b(a+ba)^*$ b) $aa^*b + (b+aaa^*b a)a^*$

گرامرهای منظم ص ۹۳ یا هر گرامر باید خطی راست باشد یا همش خطی چپ.

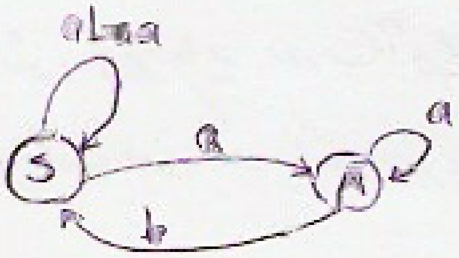
مثال: $S \to abaaS \mid aA$ حروف بزرگ ها Var هستند.

$A \to Ab \mid Aa$

$B \to b$

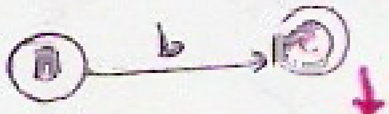
وقتی Var سمت چپ قرار بگیرد، گرامر خطی چپ است
〃 〃 〃 راست 〃 〃 راست 〃.

گرامر خطی: هر گاه هر طرفش یک Var داشته باشد. مثل این.
زبان خطی: زبانی که گرامر خطی دارد.



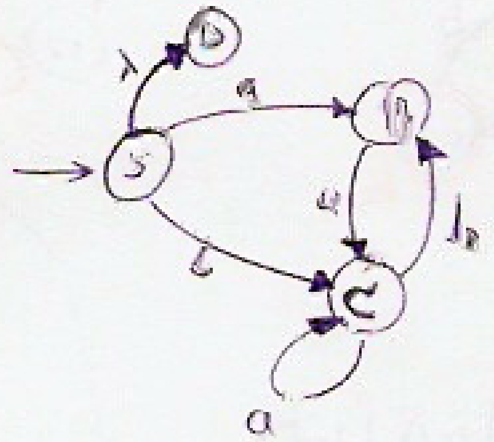
زبان منظم: زبانی که ماشین داشته باشد منظم است.

← وقتی هیچ مسیری از حالت اولیه به حالات نهایی نباشد،
می گوئیم زبان ماشین تهی است. مثل این:



حالت نهایی را اگر متغیر نداشتیم در نظر می گیریم

مثال)



$S \to abaaS \mid aA$

$A \to aA \mid bS$

$B \to b$

بجای این خط می توان نوشت:

$B \to bC$

$C \to \lambda$

$S \to aB \mid bC \mid D$

$B \to aC$

$C \to aC \mid b B$

$B \to \lambda$

$C \to \lambda$

$D \to \lambda$

قضیه) گرامر خطی راست (G)، زبان منظم ($L(G)$) تولید می کند.

قضیه) هر زبان منظم، یک گرامر منظم دارد.

قضیه) هر زبان منظم، یک عبارت منظم دارد.

قضیه) زبان L منظم است، اگر و فقط اگر یک گرامر خطی چپ G باشد که $L(G) = L$ شود.

اثبات) اگر گرامر خطی چپ بصورت

$A \to Bv$

$A \to v$

هست که برای آن می‌توان گرامر خطی راست ساخت که بصورت $A \to v^R B$ یا $A \to v^R$ باشد — این زبان چونکه گرامر خطی راست دارد منظم است. پس عکس آن هم منظم است.

مثال)

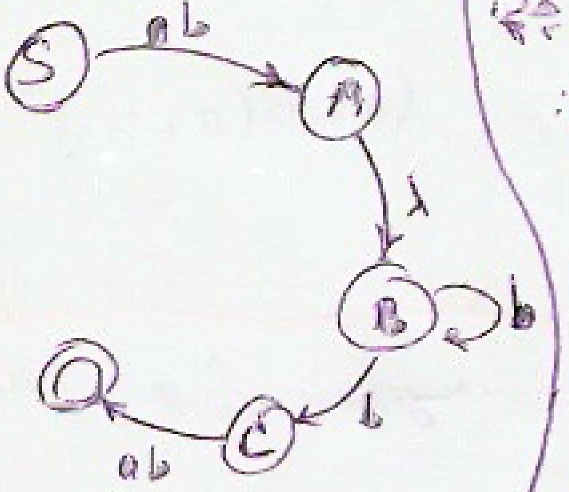
$S \to abA$

$A \to B$

$B \to bB \mid bC$

$C \to ab$

چون C با ab به هیچ کجا می‌رود حالت نهایی داریم

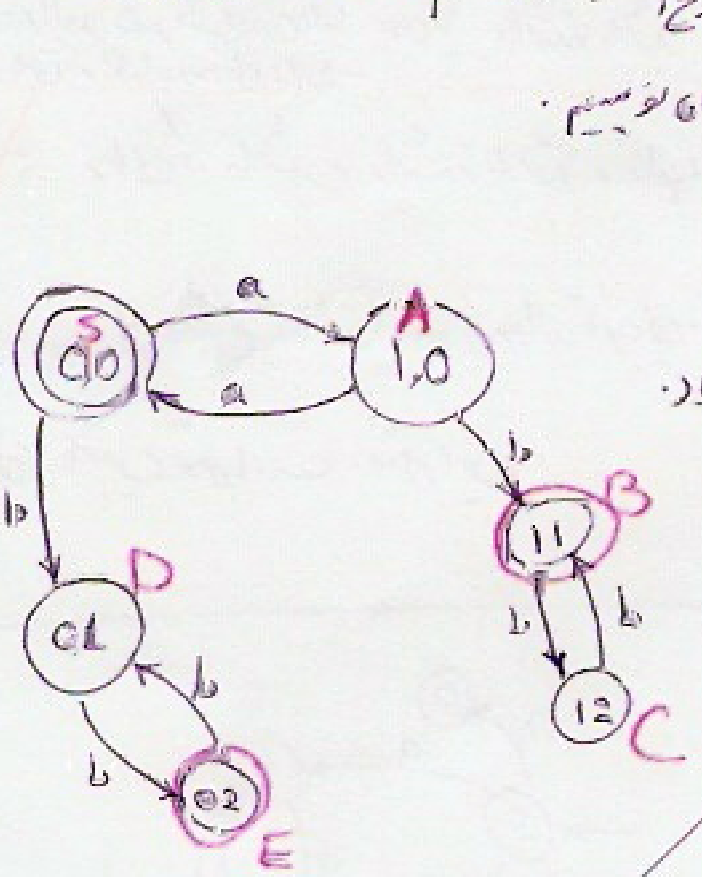


تمرین ۹ صفحه ۱۰۲: یک گرامر منظم برای زبان $L = \{a^n b^m : n+m \text{ زوج است}\}$ بنویسید.

⊖ اول ماشین آن را باید بدست آوریم و بعد گرامر آن را از روی ماشین می‌نویسیم.

⊖ $n+m$ زوج است یعنی اینکه یا هر دو زوج یا هر دو فرد هستند.

⊖ ما هم اینکه همیشه a میاد و بعدش b و بعدش دیگه a نمی‌تواند بیاید.



اینم گرامرش

$S \to \lambda \mid aA \mid bD$

$A \to aS \mid bB$

$B \to bC$

$C \to bB$

$D \to bE$

$E \to bD$

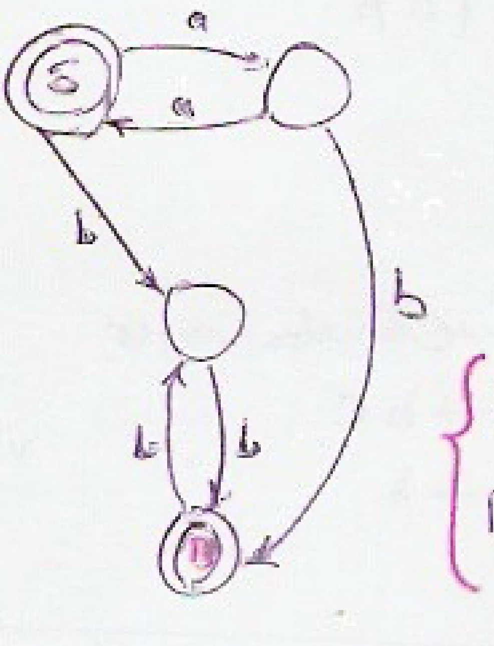
$E \to \lambda$

$B \to \lambda$

* حالا می‌توانیم ماشین را خلاصه‌تر کنیم تا گرامر کوتاه‌تری نیز بدست آوریم:

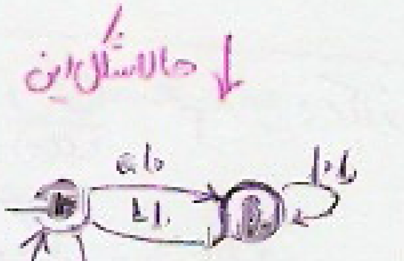
C و D یکی شوند

E و B هم یکی شوند



$$\begin{cases} S \to aaS \mid abB \mid bbB \mid \lambda \\ B \to bbB \mid \lambda \end{cases}$$

↓ حالا شکل این

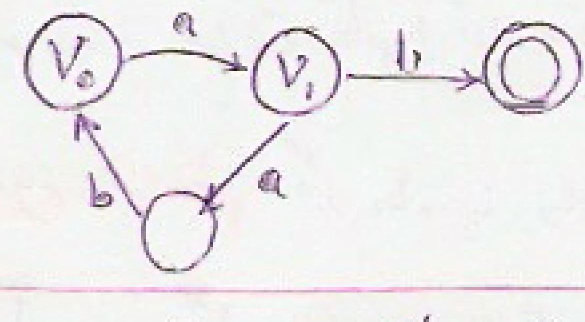


توجه:

اینو می‌شه یکجا نوشت

$S \to abB$

مثال) با گرامر
$$\begin{cases} V_0 \to aV_1 \\ V_1 \to abV_0 \mid b \end{cases}$$
یک آتاماتون بسازید!



قضیه) اگر L زبان منظم باشد، آنگاه گرامر خطی راست G برای آن وجود دارد.

مثال) برای $L(aab^*a)$ یک گرامر خطی راست بسازید

$$q_0 \to aq_1 \to aaq_2 \to aabq_2 \to aabaq_f \Rightarrow aaba$$

قضیه مهم: زبان L منظم است اگر و فقط اگر یک گرامر خطی G برای آن موجود باشد.
ص ۱۰۱ اثبات

مثال) ماشین برای رشته هایی که حداقل ۲a و دقیقا ۲b داشته باشند.

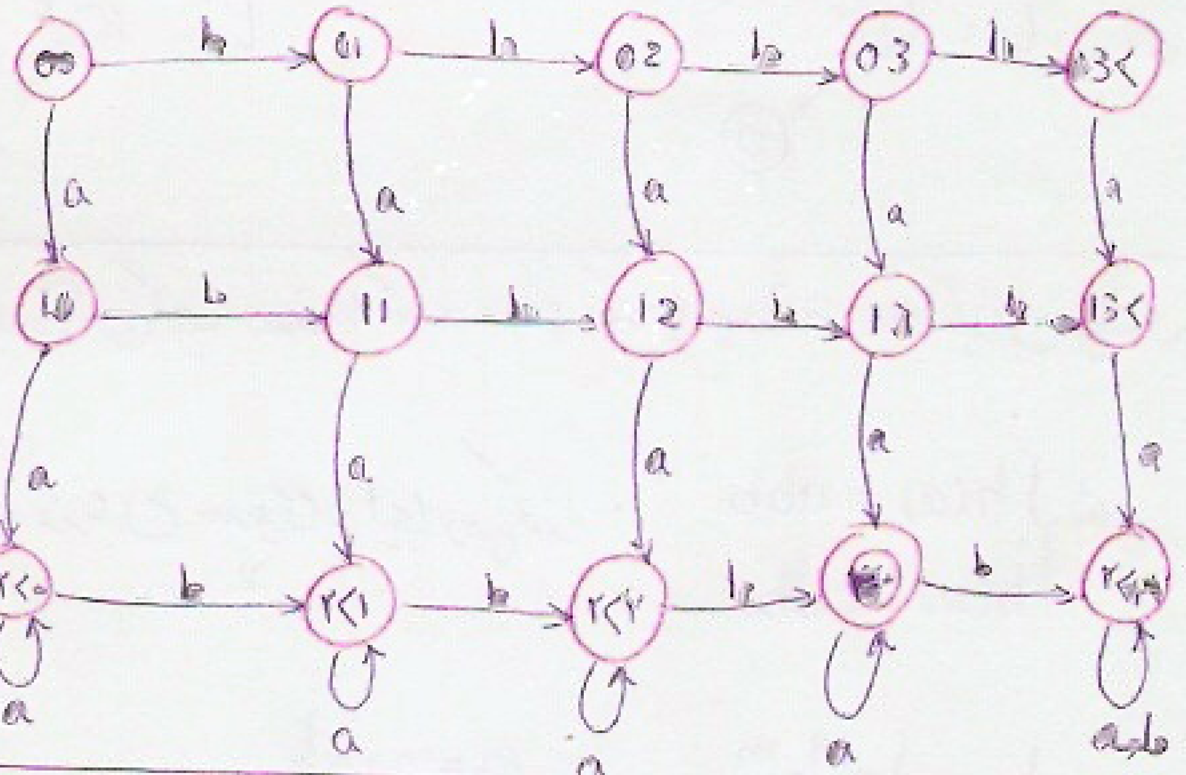
$$\begin{array}{l} a \leftarrow \{0 \quad 1 \quad 2 \\ b \leftarrow \{0 \quad 1 \quad 2 \quad 3 \end{array}$$

مثال) برای عبارت $(aab^*ab)^*$ یک گرامر خطی راست بنویسید:

$S \to aaA \mid \lambda$

$A \to bA \mid abS$

توجه: حلقه ای که در این گراف وجود دارد به دلیل وجود * در بالای پرانتز است.



$L = \{a^n b^m : n \geq 2 \ ; \ m \geq 3\}$

مثال) برای زبان روبرو گرامر خطی چپ و راست بسازید:

عبارت منظم اش را می نویسیم: aaa^*bbbb^*

حالا گرامر راست:

$S \to aaA$

$A \to aA \mid bbbB$

$B \to bB \mid \lambda$

گرامر چپ:

$S \to Abbb$

$A \to Ab \mid Baa$

$B \to Ba \mid \lambda$

زبان منظم را می‌توان توسط عبارت منظم یا گرامر منظم یا dfa نشان داد.

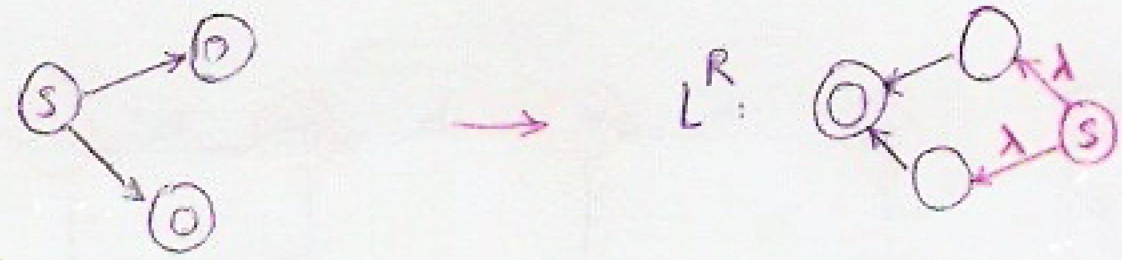
* خاصیت بسته بودن زبان های منظم: اگر L_1 و L_2 زبان های منظم باشند، آنگاه $L_1 \cup L_2$ ، $L_1 \cap L_2$ و $L_1 . L_2$ و $\overline{L_1}$ و L_1^* هم منظم هستند. (همچنین $L_1 - L_2$ منظم است چون $L_1 - L_2 = L_1 \cap \overline{L_2}$)

- اثبات متمم: اگر L_1 زبان منظم باشد، آنگاه یک dfa آن را می‌پذیرد.
آنگاه یک dfa دیگر وجود دارد که زبان $\overline{L_1}$ را می‌پذیرد پس $\overline{L_1}$ هم منظم است.

- معکوس زبان: زبان های منظم نسبت به عمل معکوس کردن نیز بسته هستند.
یعنی اگر L_1 منظم باشد، L_1^R هم منظم است.

برای اثباتش کافی است nfa برای L_1 رسم کنیم و به جای رئوس نهایی، رأس شروع بگذاریم و جای رأس شروع را هم به نهایی بدهیم و جهت یال ها را هم عکس کنیم. ماشین حاصل زبان L_1^R را می‌پذیرد. پس L_1^R هم منظم است. (اگر در L_1 چند نقطه نهایی داشتیم در ماشین معکوس، یک نقطه شروع مجازی تعریف کرده و با λ به آن نقاط می‌رویم:

مثال L_1: $\rightarrow$ L^R:



همریختی (هومومورفیک): یک جایگزینی که در آن یک حرف با یک حرف دیگر یا یک رشته، جایگزین می‌شود.

یک فانکشن مثل h روی زبان اعمال می‌شود و زبان جدیدی را ایجاد می‌کند.

مثال:
$$\begin{cases} h(a) = abb \\ h(b) = cdb \end{cases}$$

$$L_1 = \{a^n b^m : \text{زوج } m+n\}$$

$$L_2 = \{(abb)^n (cdb)^m : \text{زوج } m+n\}$$

* اگر L_1 زبان منظم باشد، هر همریختی آن نیز منظم خواهد بود.

تقسیم راست زبانها: اگر L_1 و L_2 الفبایشان مشترک باشد، خارج قسمت راست L_1 بر L_2 بصورت زیر تعریف می‌شود:

$$L_1/L_2 = \{x : xy \in L_1 \text{ برای } y \in L_2\}$$

* تمام رشته های L_1 که پسوندی متعلق به L_2 دارند در نظر می‌گیریم. هر یک از این رشته ها با حذف پسوند، متعلق به L_1/L_2 هستند.

مثال) اگر داشته باشیم:

$$L_1=\{a^nb^m : n\geq 1 , m\geq 0\}\cup\{ba\}$$

$$L_2=\{b^m \quad m\geq 1\}$$

آنگاه $L_1/L_2=\{a^nb^m \quad n\geq 1 , m\geq 0\}$

رشته های L_2 یکی یا بیشتر b ها دارند، پس b ها از L_1 برمی‌داریم تا جواب بدست آید.

$$\boxed{\text{اگر } L_1 \text{ و } L_2 \text{ منتظم باشند، آنگاه } L_1/L_2 \text{ نیز منتظم است}}$$

- زبان متناهی و نامتناهی: اگر در طول مسیر نقطه شروع تا نقطه نهایی، حلقه وجود داشته باشد، زبان نامتناهی است. در غیر اینصورت متناهی است.

نکته) زبان منتظم می‌تواند نامتناهی باشد.

- چک کردن وجود w در زبان منتظم L: زبان منتظم را بصورت dfa نشان می‌دهیم و بعد w را در آن آزمایش می‌کنیم

- تهی بودن زبان: اگر مسیر ساده‌ای در dfa زبان به هر یک از رأس های نهایی باشد، زبان غیر تهی است. در غیر اینصورت تهی است.

- تساوی دو زبان: اگر نتیجه $(L_1\cap\bar{L_2})\cup(\bar{L_1}\cap L_2)$ تهی باشد، L_1 و L_2 با هم برابرند.

- در اصل یعنی L_1-L_2 و همچنین L_2-L_1 تهی باشند، آنگاه دو زبان با هم برابرند.

تشخیص زبان های غیر منتظم) مثلاً a^nb^n که $n\geq 0$ نامنتظم است.

- زبانی منتظم است که اطلاعاتی که در هر مرحله باید بخاطر آورده شود محدود باشد.

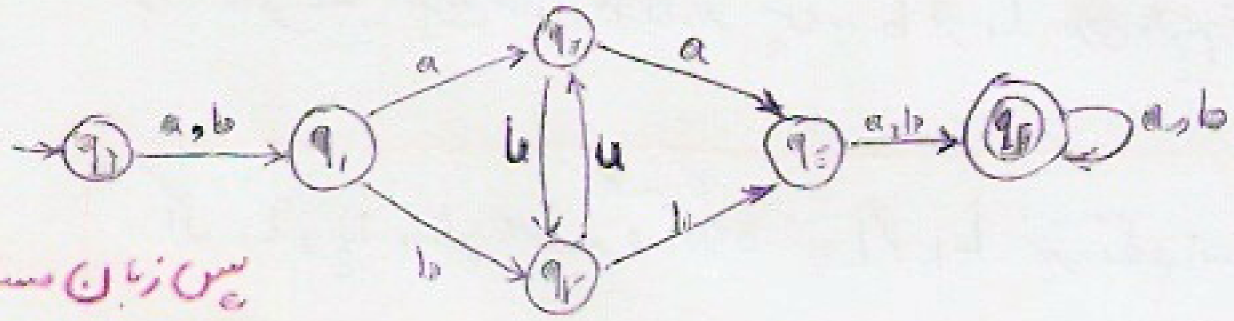
لم تزریق: در اصل می گوید که اگر در گراف n راسی، ما مسیری به طول n یا بزرگتر داشته باشیم، یک راس حداقل تکرار شده است (یعنی حلقه افتاده)

سوال) آیا L منتظم است؟ $L=\{uww^Rv : u,v,w\in\{a,b\}^+\}$

یا باید dfa آن را نشان دهیم یا از طریق لم تزریق ثابت کنیم.

ww^R ← یه چیز و معکوس آن - u,v ← هر ترکیبی از الفبا (حداقل یه کاراکتر باید باشند)

چون قبل و بعد از ww^R ما u و v را داریم، بنابراین w می تواند حتی ۱ کاراکتر باشد و عکس آن هم خودش می شود. بقیه چیزهایی هم که آمده اند را به u و v نسبت می دهیم.



پس زبان مسئله منتظم است.

چند نمونه زبان نامنظم:

$L=\{a^nb^n : n\geq 0\}$

$L=\{ww^R : w\in\{a,b\}^*\}$

$L=\{n_a(w) < n_b(w)\}$

$L=\{ww : w\in\{a,b\}^*\}$

$L=\{ww^Rv : v,w\in\{a,b\}^+\}$

$L=\{uww^Rv ; u,v,w\in\{a,b\}^+, |u|\geq|v|\}$

$L=\{a^{n!} \quad n\geq 0\}$

$L=\{a^nb^m \quad n\neq m\}$

$L=\{a^nb^m \quad n\leq m\}$

$L=\{w: n_a(w)\neq n_b(w)\}$

$L=\{a^n :$ عدد اول است $n\}$ → $\Sigma=\{a\}$

$L=\{a^{k^2} : k>0\}$

فصل ۵ (زبانهای مستقل از متن (Context Free))

گرامر مستقل از متن) در گرامر منظم سمت چپ یک متغیر و سمت راست نیز شکل خاصی داشت - اما در زبان و گرامر مستقل از متن ، سمت چپ ، یک متغیر و سمت راست هر چیزی می تواند باشد.

تعریف) گرامر $G(V,T,S,P)$ را مستقل از متن گوئیم اگر قوانین آن بصورت $A \to x$ باشد که $A \in V$ و $x \in (V \cup T)^*$ می باشد.

- همچنین یک زبان مستقل از متن است ، اگر و فقط اگر یک گرامر مستقل از متن داشته باشد.

نکته) زبان های منظم ، زیر مجموعه ای از زبان های مستقل از متن هستند.

مثال) گرامر زیر مستقل از متن است ، اما منظم نیست.

$$\begin{cases} S \to aSa \\ S \to bSb \\ S \to \lambda \end{cases}$$

مثلاً $S \Rightarrow aSa \Rightarrow aaSaa \Rightarrow aabSbaa \Rightarrow aabbaa$

زبان آن $L(G) = \{ww^R : w \in \{a,b\}^*\}$

مثال) زبان زیر مستقل از متن است. برای اثبات حرفمان باید گرامر مستقل از متنی برای آن بسازیم :

$$L = \{a^n b^m : m \neq n\}$$

در حالتی که $n \neq m$ باشد ، یعنی یا $n > m$ یا $m > n$ است.

$$S \to AS_1 \mid S_1B$$
$$S_1 \to aS_1b \mid \lambda$$
$$A \to aA \mid a$$
$$B \to bB \mid b$$

جمله : مثل aab که فقط شامل ترمینال هست ، را گویند **فرم جمله ای :** $aaBb$ که شامل ترمینال و متغیر است.

اشتقاق : رسیدن از نقطه ی شروع گرامر را تا پایان (که منجر به تولید یه جمله شود) اشتقاق گویند.

*** چپ و راست :** در قسمت جمله هایی که چند متغیر داریم ، اگر سمت چپ ترین متغیر را جایگزین کنیم ، به آن اشتقاق چپ و اگر راست ترین متغیر را با گذاشتن جایگزین کنیم اشتقاق راست می گویند.

مثلاً در این گرامر:

$S \to AB$
$A \to aaA$
$A \to \lambda$
$B \to Bb$
$B \to \lambda$

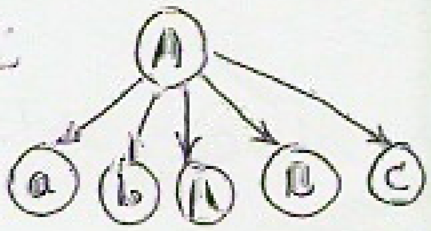
اشتقاق چپ

$$S \Rightarrow AB \Rightarrow aaAB \Rightarrow aaB \Rightarrow aaBb \Rightarrow aab$$

اشتقاق راست

$$S \Rightarrow AB \Rightarrow ABb \Rightarrow aaABb \Rightarrow aaAb \Rightarrow aab$$

درخت اشتقاق: یک درخت مرتب است که از چپ به راست پر می‌شود. مثلاً قانون $A \rightarrow abABc$ بصورت

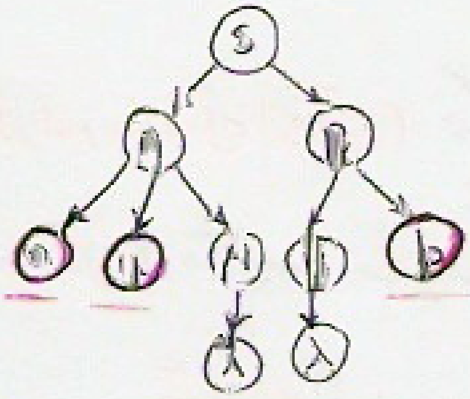


نشان داده می‌شود. برگ‌ها به ترمینال یا λ ختم می‌شوند - گره‌های غیر برگ دارای برچسب از نوع V (متغیرها).

مثال) در مثال قبل، اشتقاق چپ بصورت زیر بود

$S \Rightarrow AB \Rightarrow aaAB \Rightarrow aaB \Rightarrow aaBb \Rightarrow aab$

درخت اشتقاق آن، بصورت روبرو است:



حاصل درخت: پس از آنکه درخت رسم شد، λ ها را حذف کنید و برگ‌ها را از سمت چپ به راست بخوانید - این حاصل درخت است.

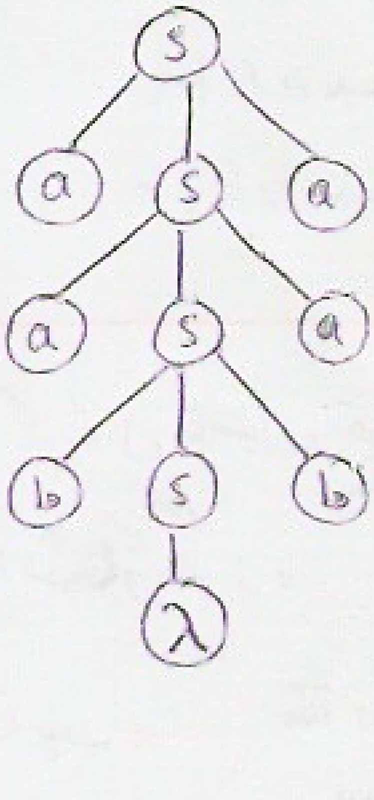
درخت اشتقاق جزئی: یعنی بخشی از یک درخت که کامل نشده و در برگ‌هایش، متغیر دارد. (V).

قضیه: اگر G یک گرامر مستقل از متن باشد، آنگاه $w \in L(G)$ یک درخت اشتقاق برای G دارد که حاصل آن w است و بالعکس. حاصل درخت اشتقاق در $L(G)$ هست.

نکته) درخت اشتقاق نشان می‌دهد که چه قوانینی از گرامر استفاده شده‌اند - اما ترتیب استفاده را نمی‌گوید.
- اگر درخت را اینگونه بگیریم که همیشه متغیر چپ توسعه یافته، یک اشتقاق چپ بدست میآوریم.

مثال) درخت اشتقاق گرامر زیر را رسم کنید:

$S \rightarrow aSa \mid bSb \mid \lambda$



رشته تولید شده

$aabbaa \Leftarrow$

گرامر مستقل از متن برای زبان‌های زیر بنویسید:

$(m,n \geq 0)$

① $L = \{a^n b^m : n \leq m+3\}$

$S \rightarrow aSbB \mid a \mid aa \mid aaa$

$B \rightarrow bB \mid \lambda$

② $L = \{a^n b^m : 2n \leq m \leq 3n\}$

$S \rightarrow aSbb \mid aSbbb \mid \lambda$

③ $L = \{a^n b^m c^{m+n}\} \xrightarrow{\text{ساده می‌کنیم}} \{a^n b^m c^m c^n\}$

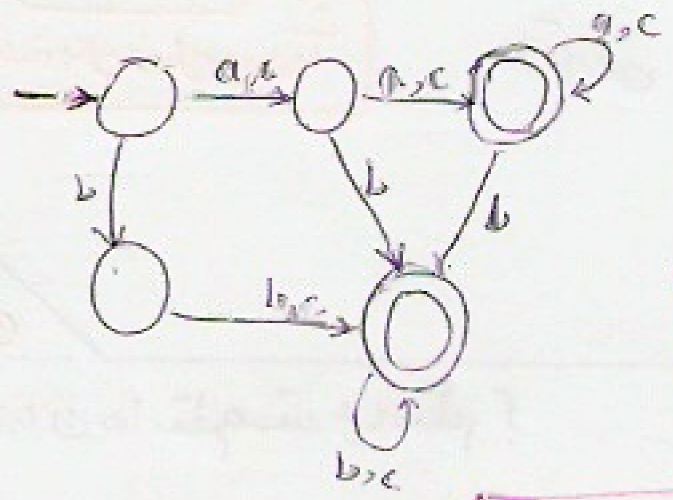
$S \rightarrow aSc \mid A$

$A \rightarrow bAc \mid \lambda$

تمرین) و حل)

$\Sigma = \{a, b, c\}$ تمام رشته هایی که حداقل طولشان ۲ بوده و تمام a ها به صورت وجود، قبل از b دیده شوند

نکته) حتما وقتی b دیده دیگه نباید a ببیند.

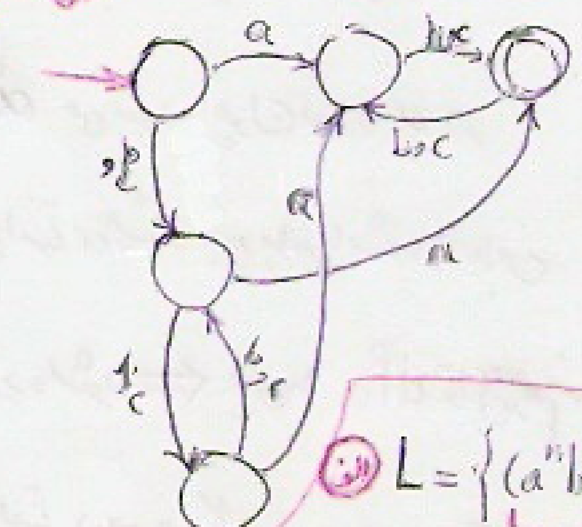


عبارت منظم

$$R = (a+c)(a+c)^* b(\lambda + (b+c)^*) + (a+c)b(\lambda + (b+c)^*) + b(b+c)^*$$

مثال) تمام رشته هایی طول زوج که شامل فقط یک a باشد. عبارت منظم آن را بنویسید. $\Sigma = \{a, b, c\}$

اول ماشین را رسم کردیم



$$R = a(b+c)\big((b+c)(b+c)\big)^* +$$
$$+ (b+c)\big((b+c)(b+c)\big)^* a \big((b+c)(b+c)\big)^* +$$
$$+ (b+c)(b+c)\big((b+c)(b+c)\big)^* a (b+c)\big((b+c)(b+c)\big)^*$$

مثال) گرامر مستقل از متن برای این زبان ها بنویسید:

(الف) $L = \{\underbrace{(a^n b^n)^k}_{S_1} \cdot \underbrace{(c^m d^m)^k}_{S_2} \quad m, n \geq 0\}$

$S \to S_1 S S_2 \mid \lambda$

$S_1 \to a S_1 b \mid \lambda$

$S_2 \to c S_2 d \mid \lambda$

توجه کنید که این زبان می تواند رشته ای مثل این $a^2b^2a^4b^4$ را تولید کند. زیرا $k=2$ است. در دفعه اول $n=2$ بوده و در دفعه دوم $n=4$.

(ب) $L = \{a^n w w^R b^n \ : w \in \Sigma^*, n \geq 1\}$

$S \to a S b \mid a S_1 b$

$S_1 \to a S_1 a \mid b S_1 b \mid \lambda$

(ج) $\{a^i b^i a^j b^j \quad i, j \geq 0\}$

راه اول

$S \to AB$

$A \to aAb \mid \lambda$

$B \to aBb \mid \lambda$

راه دوم

$S \to AA$

$A \to aAb \mid \lambda$

(د) $\{a^n b^m c^{2n+m}\} \xrightarrow{\text{ساده کردن}} a^n b^m c^m c^{2n}$

$S \to aScc \mid aAcc$

$A \to bAc \mid bc \mid \lambda$

(هـ) $L = \{a^n b^m \quad n \neq 2m\}$

دو حالت خواهد داشت $\begin{cases} n > 2m \\ n < 2m \end{cases}$

$S \to AS_1 \mid S_1 B \mid S_2 C$

$S_1 \to aaS_1 b \mid \lambda$

$A \to aA \mid a$

$B \to bB \mid b$

$S_2 \to aaS_2 \mid a$

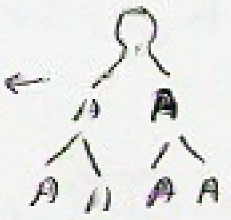
$C \to b \mid \lambda$

نکته) در گرامر به فرم چامسکی حداکثر عمق درخت n است.

این نکته مال فصل بعدی است.

حداقل عمق $\log n$ است

درخت یکطرفه رشد می‌کند. ← برای تولید n تا a ، n عمق داریم.



درخت ← وقتی متوازن رشد کند، حداقل عمق را خواهیم داشت.

تمرین

این زبان‌ها مستقل از متن هستند یا نامستقل؟

$$L_1 = \{ \alpha^i (\alpha\beta)^j (\gamma\alpha)^i \quad j \geq 0 \ , \ i \geq 0 \}$$

مستقل نیست. چون اگر $j = 0$ باشد $\alpha^i(\gamma\alpha)^i$ می‌شود. چون تعداد α در ابتدا و انتهای رشته باید برابر باشد، بنابراین باید شمرده شود ⟸ پس با PDA قابل رسم نیست.

$$L_2 = \{ \alpha^i (\alpha\beta)^j (\gamma\alpha)^{i+j} \quad j \geq 0 \ , \ i \geq 1 \}$$

$$L_3 = \{ \alpha^i (\alpha\beta)^j (\gamma\alpha)^i \ . \ j \geq 1 \text{ و } i \geq 1 \}$$

این حالت در مورد گزینه دوم هم برقرار است. یعنی اگر j را صفر بگذاریم $\boxed{\alpha}$ در ابتدا و انتها باید یکی شود و نیاز به شمارش دارد.

در مورد سومی هم اگر $j = 1$ باشد، فقط $\alpha\beta$ و $\gamma\alpha$ وسط رشته اضافه می‌شه و باز هم α های اول رشته باید با آخر رشته برابر باشه ـ پس نیاز به شمارش است. پس نامستقل است.

تمرین) گرامر مستقل از متن بنویسید :

$$L=\{a^n b^m c^k \ : k=|m-n|\}$$

k در حالت ایجاد می کند:

$M : k=m-n \longrightarrow m=k+n$

$N : k=n-m \longrightarrow n=k+m$

گرامر:

$S \rightarrow M|N$

$M \rightarrow AD$

$A \rightarrow aAb|\lambda$

$D \rightarrow bDc|\lambda$

$N \rightarrow aNc|A$

← در حالت M عبارت به صورت $a^n b^{k+n} c^k$ در می آید که با ساده سازی به $\underbrace{a^n b^n}\ \underbrace{b^k c^k}$ تبدیل می شود.

← در حالت N گرامر به صورت $a^{k+m} b^m c^k$ در می آید که با ساده سازی به $a^k \underbrace{a^m b^m} c^k$ تبدیل می شود.

تمرین) $L=\{a^n b^n \ : n\geq 0\}$ نشان دهید L^2 و L^* و L^k برای $k\geq 1$ مستقل از متن است.

حل) کافیه براش گرامر مستقل از متن بنویسیم:

برای L^2:

$S \rightarrow AA$

$A \rightarrow aAb|\lambda$

L^k:

$S \rightarrow \underbrace{AA\ldots AA}_{k}$

$A \rightarrow aAb|\lambda$

L^*:

$S \rightarrow A|SS$

$A \rightarrow aAb|\lambda$

پوشش ← یافتن یکسری قانون که با استفاده از آنها $w \in L(G)$ مشتق شود.

در اصل ساختن درخت اشتقاق برای یک w موجود در زبان است.

این پوشش مشکلاتی دارد که با حذف قوانین $A \to \lambda$ و $A \to B$ تا حدودی حل می‌شود.

قضیه) اگر G گرامری مستقل از متن باشد که قوانینی به شکل $A \to \lambda$ یا $A \to B$ نداشته باشد و $w \in T^*$ باشند، روش پوشش می‌تواند بگوید که w قابل تولید هست یا با پوشش اصلاً به دست نمی‌آید.

گرامر ساده ← گرامری را گویند که تمامی قوانین آن به صورت $A \to aX$ باشد که $A \in V$، $X \in V^*$، $a \in T$ و هر زوج (A, a) حداکثر یکبار در قوانین (P) واقع شوند. $\boxed{\text{S-Gramer}}$

مثال

$S \to aSSBA$ این خط ساده است

$A \to b|a$ " " ساده است

$B \to b|bA$ ساده نیست. چون زوج (B, b) دو بار دیده می‌شوند.

گنگی: گرامر G را گنگ گوئیم، اگر برای یک $w \in L(G)$ حداقل دو درخت اشتقاق متفاوت داشته باشیم.

گنگی را در زبان های برنامه سازی، با اولویت دادن به $*$ نسبت به $+$ حل می‌کنند. البته بهتر آن است که گرامر را طوری بازنویسی کنیم که فقط یک پوشش برای w ممکن باشد.

زبان غیرگنگ: اگر L زبان مستقل از متن باشد که برای آن یک گرامر غیرگنگ وجود داشته باشد، آنرا زبان غیرگنگ گویند.

نکته) هر گرامر ساده، غیرگنگ است.

تمرین: یک گرامر ساده برای $L(aaa^*b + b)$ بنویسید:

$S \to aA|b$

$A \to aB$

$B \to aB|b$

تمرین صفحه ۱۶۰ سوال ۱۳: ثابت کنید گرامر روبرو گنگ است. $S \to aSb|SS|\lambda$

رشته ab را مثال می‌زنیم که ۲ اشتقاق دارد.

$$\begin{cases} S \Rightarrow aSb \Rightarrow ab \\ S \Rightarrow SS \Rightarrow S \Rightarrow aSb \Rightarrow ab \end{cases}$$

* گرامرها را در این فصل محدود به قوانینی می کنیم تا ساده تر شده و سودمندتر گردند - ترجیح این است که λ را حذف کرده و به زبان هایی بپردازیم که λ در آنها نباشد.

یک جایگزینی ساده: فرض کنید داریم
$$\begin{cases} A \to aCb \\ C \to f \mid h \end{cases}$$
کافی است C را در داخل خط اول جایگزین کنیم و خط دوم را حذف نماییم $A \to ahb \mid afb$. توجه کنید که متغیر C نباید با A یکی باشد و همچنین C باید قدم بعد بدست بیاد.

متغیر بی فایده ← در گرامر G ، متغیر A مفید است اگر و فقط اگر یک w را مشتق کند حداقل - بصورت
$$\boxed{S \stackrel{*}{\Rightarrow} xAy \stackrel{*}{\Rightarrow} w}$$
در غیر اینصورت بی فایده است و قانونی که متغیر بی فایده دارد، قانون بی فایده است.

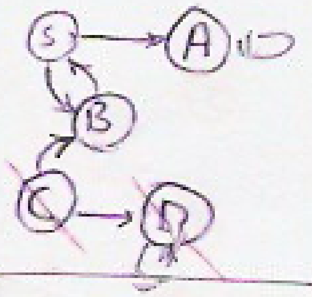
تشخیص متغیر بی فایده ← ① نتوان از متغیر شروع به آن رسید ② نتواند رشته ترمینال تولید کند.

$S \to A$
$A \to aA \mid \lambda$
$B \to bA$
در اینجا از S به B راهی نیست پس B بی فایده است

نحوه شناسایی متغیرهایی که از S به آنها دسترسی نیست:
برای اینکار کافی است گراف وابستگی را برای گرامر رسم کنید - رئوسی که از S راهی برای ورود به آنها نیست، غیر مفید هستند و باید حذف شوند.

نحوه رسم گراف وابستگی:
$A \to aBp$
فقط متغیرها مهم هستند و رسم می شوند
$A \to B$

مثال
$S \to aA \mid bB$
$A \to aA \mid a$
$B \to aS$
$C \to aB \mid D$
$D \to dD$



D و C غیر مفیدند.

مثال
$S \to aS \mid A \mid C$
$A \to a$
$B \to aa$
$C \to aCb$

در این مثال B و C بی فایده اند.
چون از S به B نمی توان رسید.
و C هرگز به ترمینال نمی رسد.

قضیه) به ازای هر گرامر مستقل از متن G ، یک گرامر معادل وجود دارد، قوانین بی فایده نداشته باشد.

قوانین λ : قانونی که سمت راست آن تهی است ($A \to \lambda$) را قانون λ گویند.

هر متغیری که با چند اشتقاق به λ برسد هیوا است. $A \overset{*}{\Rightarrow} \lambda$

نحوه حذف قوانین λ : متغیرهای هیوا را شناسایی کن. در گرامر به جای متغیرهای هیوا، λ بذار و بعد قوانین λ را حذف کن

مثال)
$$S \to Aa \mid Bb$$
$$B \to AC$$
$$A \to F$$
$$\cancel{F \to \lambda}$$
$$C \to M$$
$$\cancel{M \to \lambda}$$

$\Longrightarrow$

$$\cancel{C \to \lambda}$$
$$\cancel{A \to \lambda}$$
$$B \to AC \to \lambda$$
$$S \to a \mid b$$

$\Longrightarrow$ $S \to a \mid b$

در این مثال همه متغیرها بجز S هیوا بودند.

مثال)
$$S \to aS_1b$$
$$S_1 \to aS_1b \mid \lambda$$

$\Longrightarrow$

$$S \to aS_1b \mid ab$$
$$S_1 \to aS_1b \mid ab$$

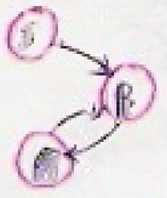
گرامر روبرو یک زبان بدون λ بنام $\{a^n b^n \mid n \ge 1\}$ تولید می‌کند. در اینجا با افزودن ab، قانون λ را در S_1 حذف کردیم.

قوانین یکه : قوانینی که بصورت $A \to B$ هست که $A,B \in V$ هستند را گویند.

قضیه) هر گرامر مستقل از متن، یک گرامر معادل خالی از قوانین یکه دارد.

حذف قوانین یکه) گراف وابستگی یکه را رسم کن و از روی آن قوانینی مثل B را اگر $B \overset{*}{\Rightarrow} A$ پیدا کن - بعد مقادیر A را در قانون فوق بگذار و بعد $B \to A$ را حذف کن!

مثال)
$$S \to aS \mid bA \mid B$$
$$A \to \cancel{B} \mid m$$
$$B \to \cancel{A} \mid d$$



$\longrightarrow$

$$S \to aS \mid bA \mid d \mid m$$
$$A \to m \mid d$$
$$B \to d \mid m$$

توجه) در گراف وابستگی یکه، فقط وقتی متغیری، مستقیم به متغیر دیگر می‌رود، آن را رسم می‌کنیم. $S \to bA$ هم رسم نمی‌شود

قضیه) هر زبان مستقل از متن بدون لاندا، گرامر مستقل از متنی دارد که هیچ قانون λ، قانون‌های بیهوده و یکه ندارد.

نکته) زبان بدون λ یعنی زبانی که S به λ نمی‌رسد.

مثال
$$S \to Aa \mid a$$
$$A \to \lambda$$

زبان گرامر روبرو λ ندارد. امّا گرامرش قانون λ دارد.

فرم نرمال چامسکی) گرامری که قوانین آن بصورت $A \to BC$ یا $A \to a$ باشد فرم چامسکی است.

دقیقاً دو متغیر — دقیقاً یک ترمینال

> **قضیه)** هر گرامر مستقل از متن که در آن λ جزء زبان گرامر نباشد دارای گرامر معادل چامسکی است.

مثال) به چامسکی تبدیل کن

$$S \to ABa \qquad \xrightarrow{\text{مرحله اول}} \qquad S \to ABB_a \qquad \xrightarrow{\text{حالا}} \qquad S \to AD_1$$

$$A \to aab \qquad\qquad A \to B_aB_aB_b \qquad\qquad D_1 \to BB_a$$

$$B \to Ac \qquad\qquad B \to AB_c \qquad\qquad A \to B_aD_2$$

$$\left.\begin{aligned} B_a &\to a \\ B_b &\to b \\ B_c &\to c \end{aligned}\right\} \text{اینها هم} \qquad D_2 \to B_aB_b$$

$$B \to AB_c$$

برای تبدیل به چامسکی، گرامر نباید قوانین λ یا یکه داشته باشد. اگر داشت باید اول اونا رو حذف کرد.

فرم نرمال گریباخ. گرامری گریباخ است که قوانین آن بصورت $A \to aX$ باشد که a ترمینال و $X \in V^*$ است.

> **قضیه)** هر گرامر مستقل از متن که λ در زبان آن نباشد دارای گرامر معادل در فرم گریباخ است.

مثال: اینو گریباخ کن:

$$S \to abSb \mid aa$$

متغیرهای جدید تعریف می‌کنیم و جای b سمت راست و a سمت راست می‌ذاریم:

$$\begin{cases} S \to aBSB \mid aA \\ A \to a \\ B \to b \end{cases}$$

مثال) گرامر $S \to aSb \mid aS \mid aaS$ را گریباخ کنید:

$$S \to aSS_2 \mid aS \mid aS_1S$$

$$S_1 \to a$$

$$S_2 \to b$$

مثال) قوانین λ ولی فایده را در این گرامر حذف کنید:

$$\begin{cases} S \to AaB \mid aaB \\ A \to \lambda \\ B \to bbA \mid \lambda \end{cases}$$

حل) اول قوانین λ را حذف کن و بعد بی فایده.

$$\begin{cases} S \to aB \mid aaB \mid aa \mid a \\ B \to aa \end{cases}$$

در اینجا توجه کنید که چون B خودش به λ هم می رفت بنابراین دو حالت اولیهٔ S بدون B هم ایجاد می شود

نکته) گاهی حذف قوانین λ باعث ایجاد قوانین یکه ای می شود که از اول در گرامر نبوده اند:

$$\begin{array}{l} S \to AB \mid ab \\ A \to aAb \mid \lambda \\ B \to b \end{array} \quad \xrightarrow[\text{قوانین } \lambda]{\text{بعد از حذف}} \quad \begin{array}{l} S \to AB \mid ab \mid B \\ A \to aAb \mid ab \\ B \to b \end{array}$$

مثال) گرامر $\begin{cases} S \to aSaA \mid A \\ A \to abA \mid b \end{cases}$ را به فرم چامسکی درآورید.

توجه) اول باید در گرامر فوق، قانون یکهٔ $S \to A$ را حذف کنید. و خط اولش تبدیل به $S \to aSaA \mid abA \mid b$ می شود.

حل)

$$\begin{cases} S \to A_1SA_1A \mid A_1A_2A \mid b \\ A \to A_1A_2A \mid b \\ A_1 \to a \\ A_2 \to b \end{cases}$$

حالا متغیرها را در سمت راست دوتایی می کنیم:

$S \to A_1A_3$

$A_3 \to SA_4$

$A_4 \to A_1A$

$S \to A_1A_5$

$A_5 \to A_2A$

$A \to A_1A_5$

$A \to b$

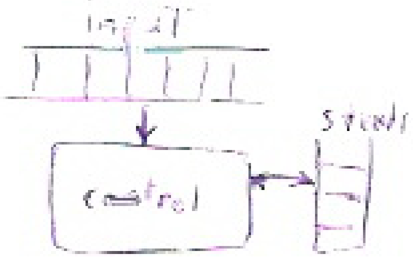
$A_1 \to a$

$A_2 \to b$

$S \to b$

فصل ۷ آتاماتای پشته‌ای

Pda ← ماشین هایی هستند که توانایی تشخیص زبان های مستقل از متن را دارند و دارای پشته هستند



در آتاماتای پشته‌ای، هر حرکت واحد کنترل، یک ورودی و علامت بالای stack اخذ می‌شود. نتیجه این حرکت، یک وضعیت جدید برای control unit و یک تغییر در پشته است.

nPda : آتاماتای پشته‌ای نامعین: به صورت $M=(Q,\Sigma,\Gamma,\delta,q_0,z,F)$ تعریف می‌شود.

(گاما بزرگ)

Q ← وضعیت ها، Σ ← الفبا، Γ ← الفبای پشته، δ ← تابع انتقال، q_0 ← وضعیت اولیه، z ← علامت شروع پشته، F ← وضعیت های نهایی

مثال

$\delta(q_1,a,b)=\{(q_2,cd),(q_3,\lambda)\}$

(q_1: وضعیت، a: ورودی، b: بالای پشته)

← یعنی اگر آتاماتا در وضعیت q_1 باشد و در ورودی مقدار a را ببیند و در بالای پشته b باشد به یکی از حالات روبرویش می رود:

(۱) به q_2 می رود و cd را جایگزین b در پشته می کند.

(۲) به q_3 می رود و b را از پشته برمی دارد اما چیزی به جایش نمی گذارد.

> * اگر λ بجای ورودی بیاید یعنی ماشین بدون مصرف ورودی به حالت دیگری می رود. مثل $\delta(q_1,\lambda,b)$

- وقتی در npda از یک حالت به حالت دیگر می خواهیم برویم، بین دو حالت علامت $\vdash$ می گذاریم.
- اگر با چند تغییر حالت، از یه حالت اولیه به یه حالت دیگه برسیم، بین آنها $\vdash^*$ می گذاریم.

پذیرفته شدن در ماشین آتاماتای پشته‌ای ← یعنی بعد از اتمام ورودی، ماشین در حالت نهایی متوقف شود.

مثال) برای زبان $L=\{n_a(w)=n_b(w)\}$ یک npda بسازید:

روش حل: در این مثال هر وقت a دیدیم آن را وارد پشته می‌کنیم - هرگاه b دیدیم یک دونه از پشته برمی‌داریم. حال اگر تعداد a تمام شد و b داشتیم، b را وارد پشته می‌کنیم و با دیدن a یک دونه b از پشته در می‌داریم.

حالت نهایی

$\delta(q_0,\lambda,z)=(q_f,z)$

$\delta(q_0,a,z)=(q_0,0z)$

$\delta(q_0,b,z)=(q_0,1z)$

$\delta(q_0,a,0)=(q_0,00)$

$\delta(q_0,b,0)=(q_0,\lambda)$

$\delta(q_0,a,1)=(q_0,\lambda)$

$\delta(q_0,b,1)=(q_0,11)$

در مثال روبرو، با دیدن a، کاراکتر 0 وارد پشته می‌شود.
و با دیدن b، کاراکتر 1 " " "

مثلاً رشته baab را با آتاماتای فوق امتحان می‌کنیم:

$$(q_0,baab,z)\vdash(q_0,aab,1z)\vdash(q_0,ab,z)\vdash(q_0,b,0z)\vdash(q_0,\lambda,z)\vdash(q_f,\lambda,z)$$

مثال) برای زبان $\{w\in\{a,b\}^+ : ww^R\}$ یک npda بسازید:

ww^R ← یعنی یک رشته + معکوسش.

باید تا نصفه رشته، آن را در پشته push کنیم و از نصفه به بعد pop کنیم - اما ما نمی‌دانیم وسط رشته کجاست - اما npda با خاصیت نامعینی این مشکل را حل می‌کند.

$\delta(q_0,a,z)=(q_0,az)$

در ورودی a است و بالای پشته z است. a را به پشته اضافه می‌کند.

$\delta(q_0,b,z)=(q_0,bz)$

$\delta(q_0,a,a)=(q_0,aa)$

$\delta(q_0,a,b)=(q_0,ab)$

$(q_0,b,b)=(q_0,bb)$

$(q_0,b,a)=(q_0,ba)$

حالات فوق، برای وارد کردن ورودی در پشته است.

$\delta(q_0,\lambda,a)=(q_1,a)$

$\delta(q_0,\lambda,b)=(q_1,b)$

در دو حالات فوق، بدون مصرف ورودی، تغییر وضعیت می‌دهیم. یعنی وسط رشته را یافته‌ایم

$\delta(q_1,a,a)=(q_1,\lambda)$

$\delta(q_1,b,b)=(q_1,\lambda)$

$\delta(q_1,\lambda,z)=(q_f,z)$

اینجا از وسط رشته رد شدیم و از این به بعد با دیدن a و b فقط pop می‌کنیم.

۳۱

تمرین: اول تبدیل به گریباخ کنید و بعد ماشین بکشید.

$S \rightarrow aBaBAb$

$B \rightarrow bB \mid a$

$A \rightarrow bSA \mid a$

$D \rightarrow a$

$C \rightarrow b$

حل $\Longrightarrow$

گریباخ

$S \rightarrow aBDBAC$

$B \rightarrow bB \mid a$

$A \rightarrow bSAD$

$D \rightarrow a$

$C \rightarrow b$

$\delta(q_0, \lambda, z) = \{(q_1, Sz)\}$

$\delta(q_1, a, S) = \{(q_1, BDBAC)\}$

$\delta(q_1, b, B) = \{(q_1, B)\}$

$\delta(q_1, a, B) = \{(q_1, \lambda)\}$

$\delta(q_1, b, A) = \{(q_1, SAD)\}$

$\delta(q_1, a, D) = \{(q_1, \lambda)\}$

$\delta(q_1, b, C) = \{(q_1, \lambda)\}$

$\delta(q_1, \lambda, z) = \{(q_f, \lambda)\}$

تمرین: مثالی برای فرم چامسکی:

$S \to aSb \mid ab$ $\equiv$ $\begin{cases} S \to AS' \mid AB \\ S' \to SB \\ A \to a \\ B \to b \end{cases}$ فرم چامسکی

هر کدام را یک متغیر در نظر می‌گیریم. A و B

ص ۱۸۹ تمرین ۶/۶ ، مثال فوق را تبدیل به گریباخ می‌کنیم.

$\begin{cases} S \to aSB \mid aB \\ B \to b \end{cases}$

مثال دیگر:

$S \to AbB \Rightarrow S \to aB'bB$

$A \to ab \Rightarrow A \to aB'$

$B \to d$ $\quad B' \to b$

ترکیب آتاماتای پشته‌ای:

مثال $\delta(q, a, b) = \{(q_1, cd)\}$

یعنی c باید بالای استک بیفته، اول d رو بعد c وارد پشته می‌شن.

مثال $\delta(q, \lambda, b) = \{(q, cb)\}$

در اینجا b اول حذف میشه بعدش cb ، push میشه. یعنی c به استک اضافه شده.

مثال $\delta(q, a, b) = \{(q, cb), (q_2, fff)\}$

در اینجا یعنی شبیه غیرقطعی است. هم به q_1 می‌رود و هم به q_2.

تمرین) گرامر مستقل از متن بنویسید.

$\{a^k b^m c^n : k = m \text{ or } m = n\}$

$\boxed{k=m}$

$S \to S_1 \mid S_2$

$S_1 \to S_1 c \mid T$

$T \to aTb$

$T \to \lambda$

$\boxed{m=n}$

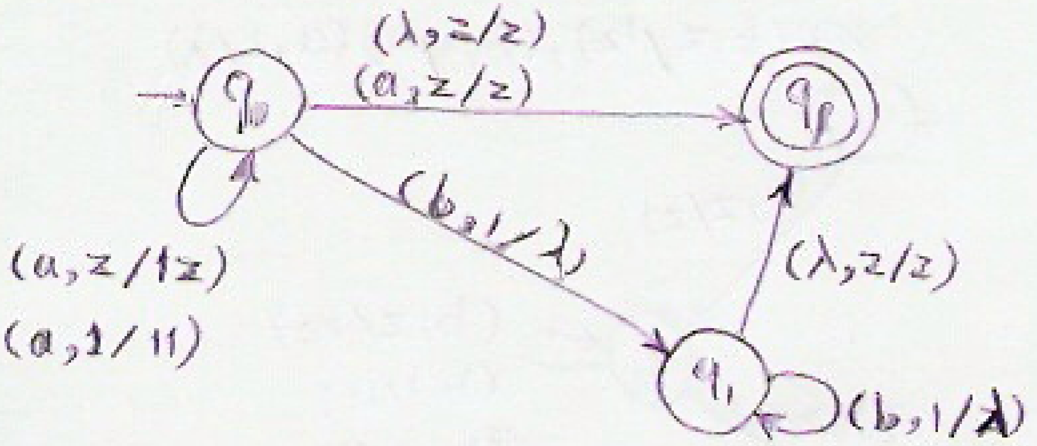
$S_2 \to aS_2 \mid B$

$B \to bBc$

$B \to \lambda$

در اینجا دو حالت داریم و برای هر حالت، گرامر جدا می‌نویسیم و در خط اول $S \to S_1 \mid S_2$ می‌توانیم وارد هر دو گرامر شویم.

مثال) یک npda با کمتر از ۴ وضعیت بیابید که زبان $L=\{a^n b^n : n\geq 0\}\cup\{a\}$ را بپذیرد.



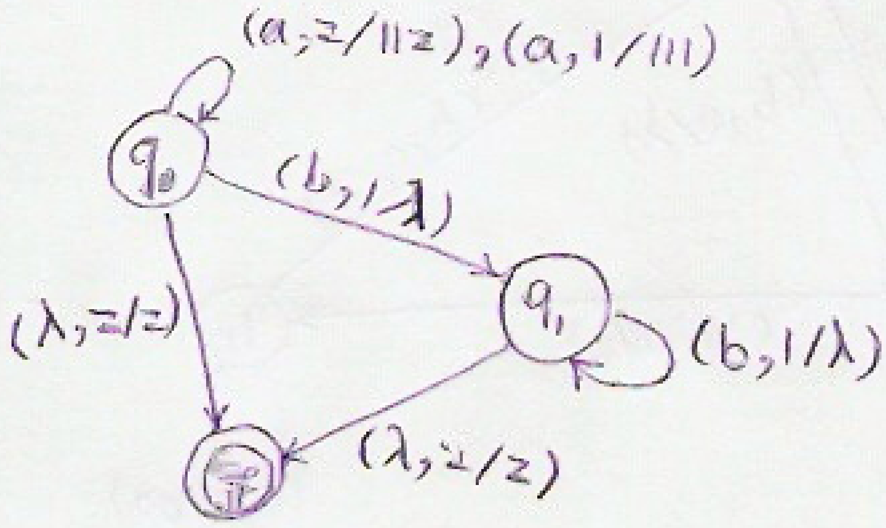
در این مدل از نمایش اول مقدار ورودی و بعد مقدار داخل پشته را که ماشین pop می‌کند و بعد از / مقداری که باید در پشته push شود را می‌نویسیم:

$(a, z / 1z)$

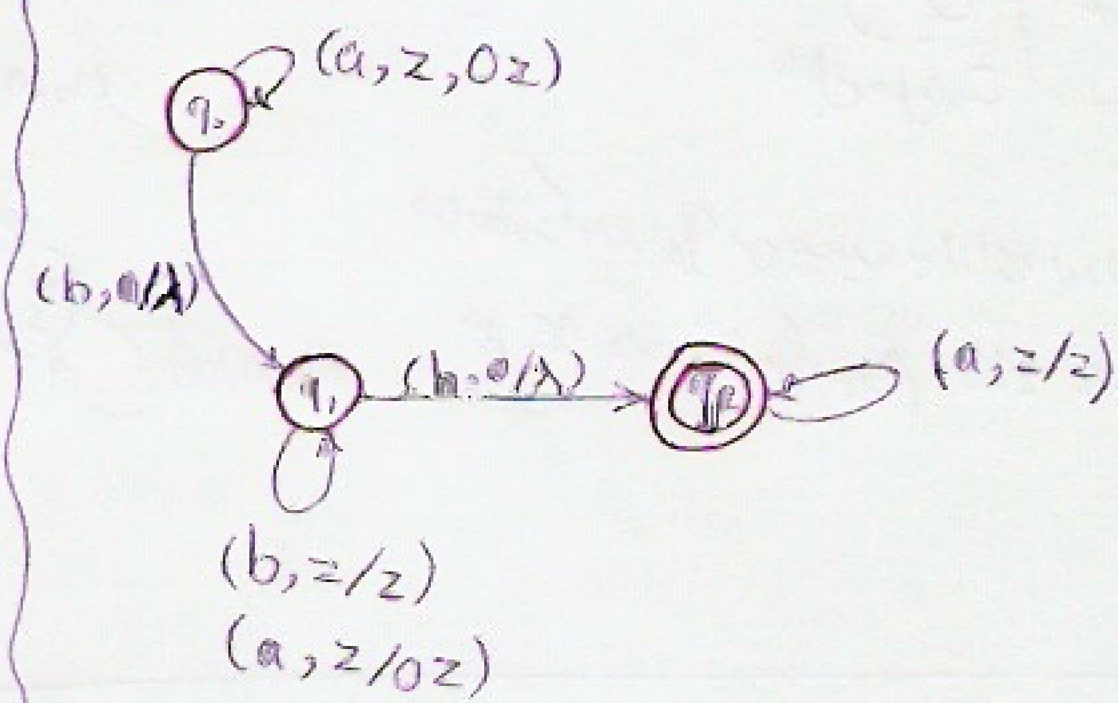
ورودی — پشته pop شود — اینو push کن

اینکه به کدام وضعیت می‌رویم هم توسط یال‌ها نشان داده می‌شود. ←

مثال) npda ای بکشید $L=\{a^n b^{2n} : n\geq 0\}$



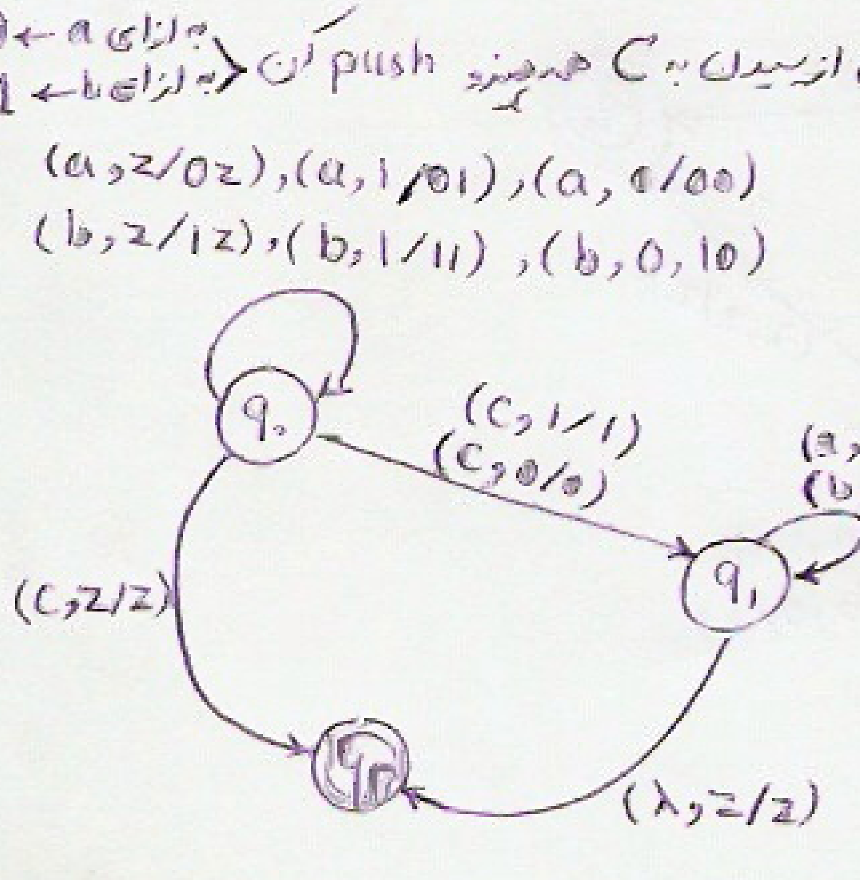
مثال) npda بکشید: $L(abb^*aba^*)$



مثال) $L=\{wcw^R : w\in\{a,b\}^*\}$

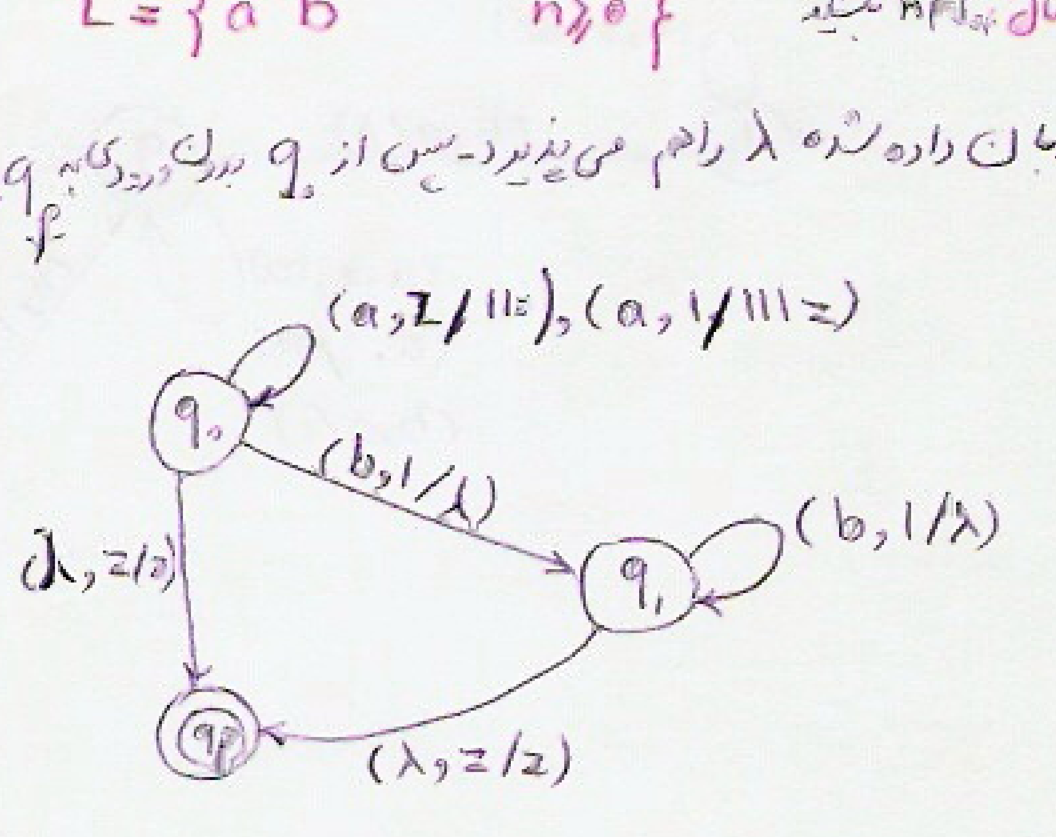
تا قبل از رسیدن به c همه چیز را push کن: به ازای a ← 0 ، به ازای b ← 1

$(a, z/0z), (a, 1/01), (a, 0/00)$
$(b, z/1z), (b, 1/11), (b, 0, 10)$



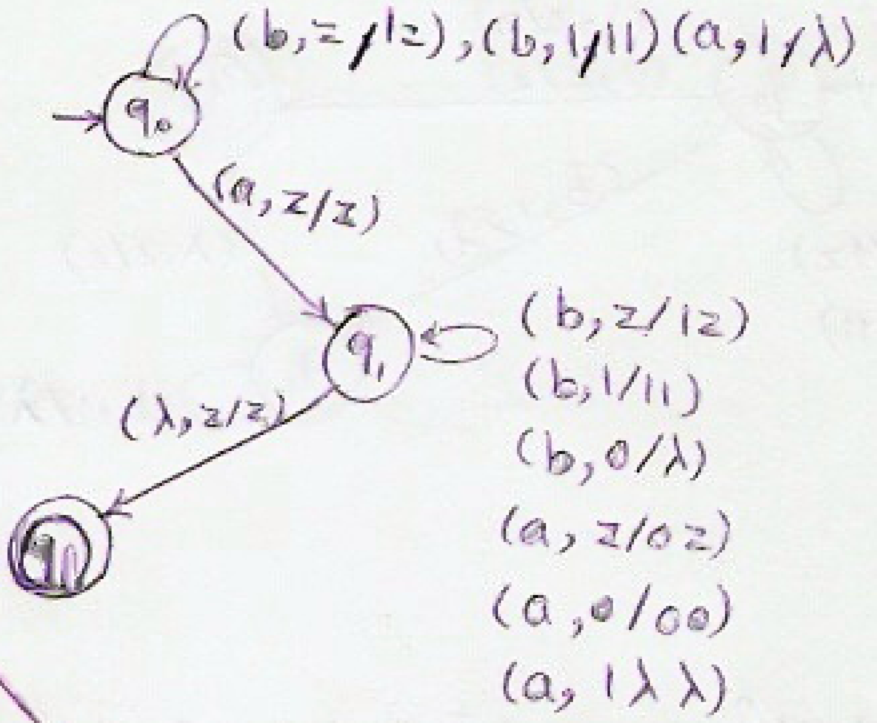
مثال) npda ای بکشید $L=\{a^n b^{2n} \quad n\geq 0\}$

زبان داده شده λ را هم می‌پذیرد، پس از q_0 بدون ورودی به q_f می‌رود



مثال) $L=\{n_a(w)=n_b(w)+1\}$

تعداد bها یکی کمتر از aها است.



در مرحله اول به ازای هر b ← مقدار 1 وارد پشته شده

و به ازای هر a ← یه دونه 1 از پشته pop می‌کنه

ولی اگر رشته با a پدیدار شد و z باشد به q_1 می‌رود

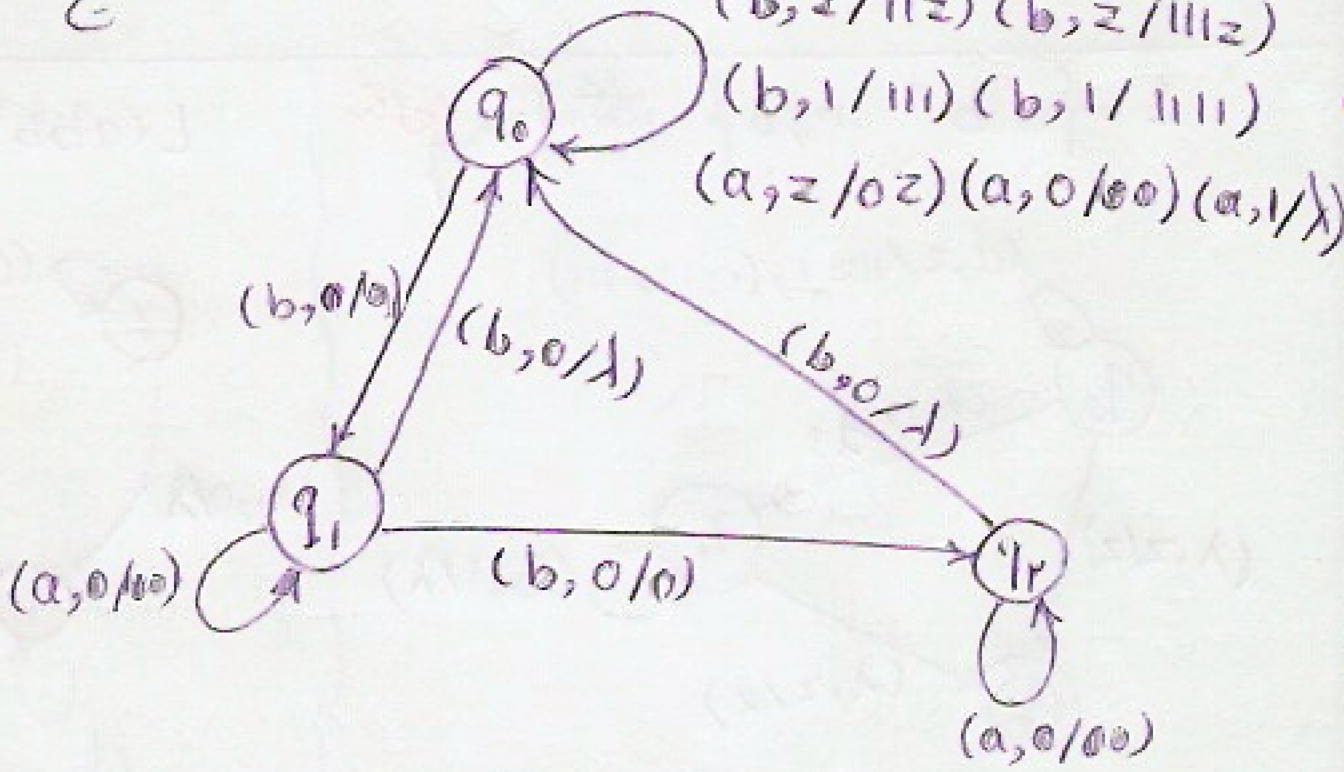
و بقیه تست‌ها را در q_1 انجام می‌دهد.

مثال

$L=\{w:\ 2n_a(w)\leq n_b(w)\leq 3n_a(w)\}$

توضیح ← این یعنی به ازای هر b، یا ۲ a یا ۳ a می‌تواند وجود داشته باشد

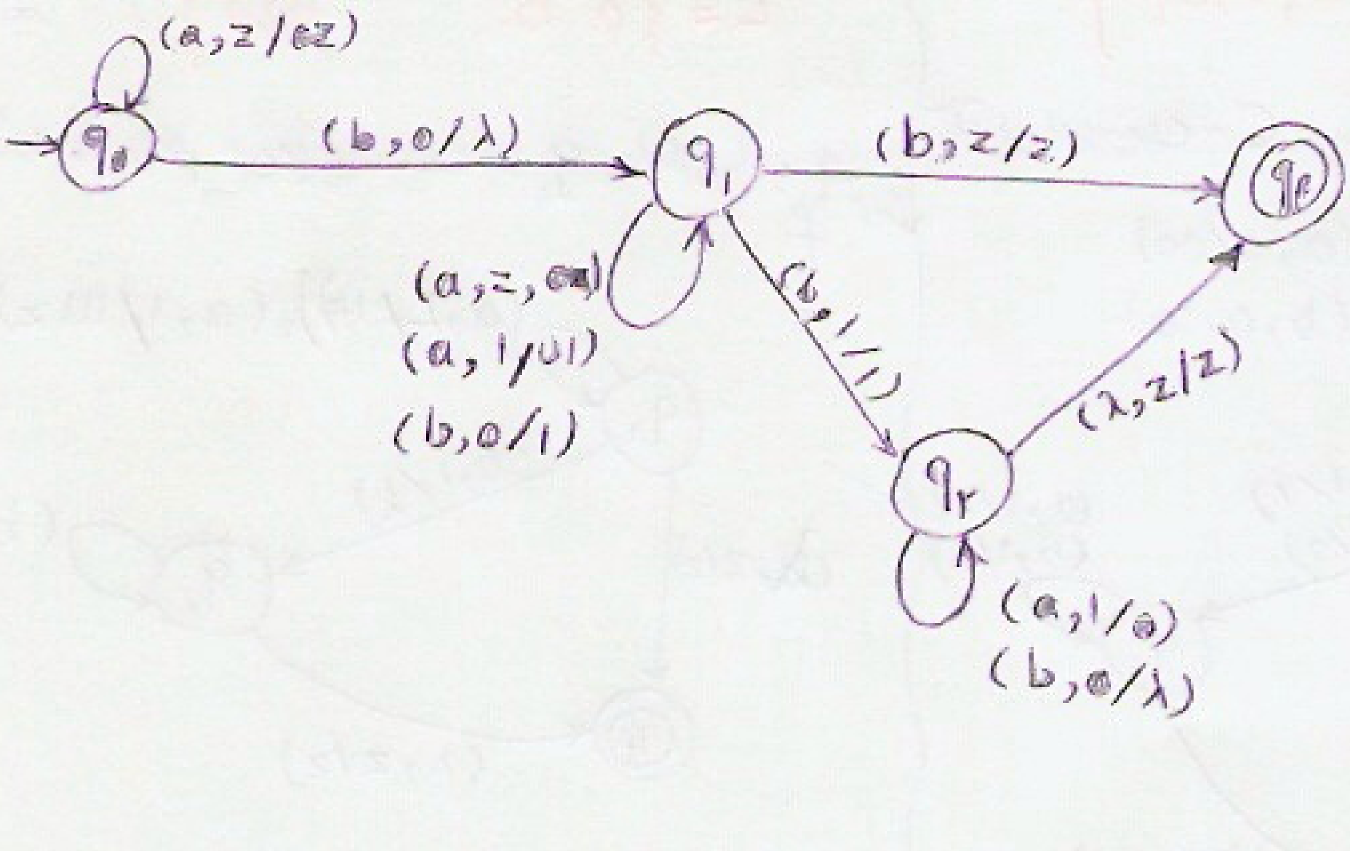
بنابراین با استفاده از آتاماتای نامعین قابل حل است



همانطور که در q_0 می‌بینید، به ازای ورودی b هم 11 وارد پشته می‌کنیم و هم 111 را.

مثال) رسم کنید npda:

$L=\{ab(ab)^n\, b\, (ab)^n\ :\ n\geq 0\}$



رابطهٔ npda و زبان های CF :

- برای هر زبان مستقل از متن، یک npda وجود دارد و برعکس هر زبانی که توسط npda پذیرش شود، CF است.

تولید آتاماتای پشته ای برای گرامر های CF :

ابتدا گرامر را به فرم گریباخ در می آوریم سپس اولین متغیر شروع را که S است بدون مصرف ورودی در پشته می گذاریم. سپس برای تبدیل $A \to aX$
- A باید در بالای پشته داشته باشیم و pop کنیم.
- a باید در ورودی باشد.
- X متغیر یا متغیرهایی است که push می شود.

مثال) گرامر $S \to aSbb|a$ را برای آن pda بسازید.

اول باید آن را به فرم گریباخ درآوریم:

سرپشته ← ورودی ← push

$$\begin{cases} S \to aSA \mid a \\ A \to bB \\ B \to b \end{cases}$$

این برای شروع است: $\delta(q_0, \lambda, z) = \{(q_1, Sz)\}$

خط اول گرامر: $\delta(q_1, a, S) = \{(q_1, SA), (q_1, \lambda)\}$

خط دوم ": $\delta(q_1, b, A) = \{(q_1, B)\}$

خط سوم ": $\delta(q_1, b, B) = \{(q_1, \lambda)\}$

خط نهایی: $\delta(q_1, \lambda, z) = \{(q_f, \lambda)\}$

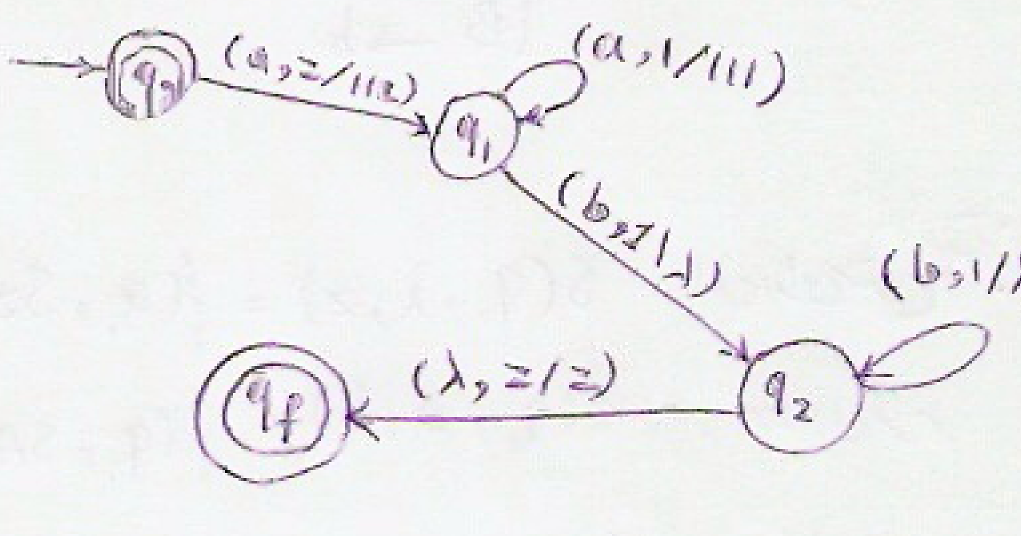
آتاماتای پشته‌دار معین (dPdA): یک آتاماتای پشته‌ای است که در انجام حرکتش انتخابی ندارد
(مثلاً برای $\delta(q, a, b)$ فقط یک وضع وجود دارد که بره بهش) — و اگر با $\delta(q, \lambda, b)$ ماشین تغییر
حالت می‌دهد دیگه با $(q, \underset{\downarrow}{c}, b)$ تغییر حالت نمی‌دهد
حرف دیگری

نکته) برای معین بودن dPdA شرط دیگری وجود ندارد - مثلاً می‌توانیم در آن وضعیت تهی یا مرده داشته باشیم.

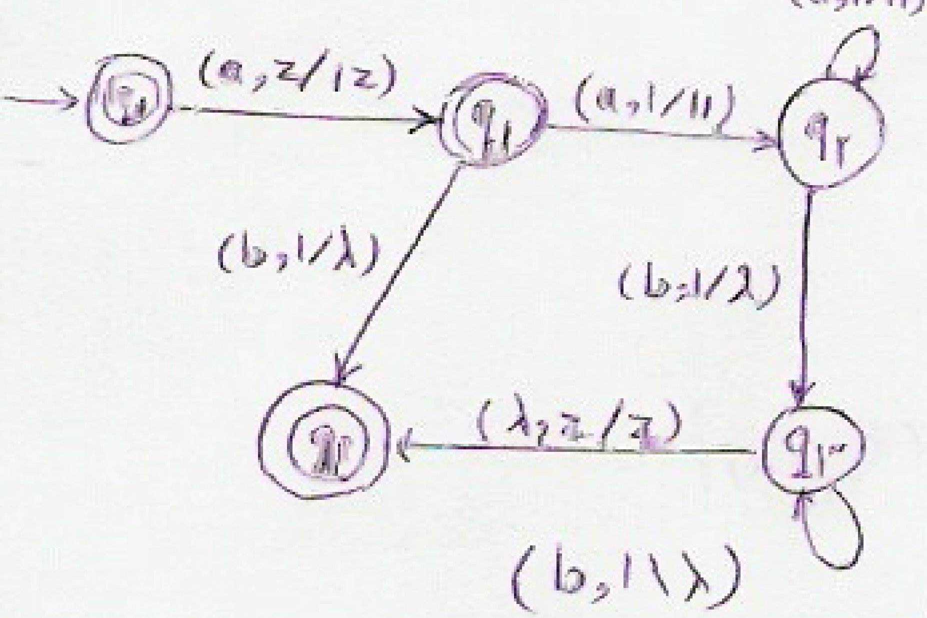
زبان مستقل از متن معین: اگر زبانی دارای یک dpda باشد که آن را بپذیرد، مستقل از متن معین است.

مثال) wcw^R ← قطعی است، چون بعد از c می‌دانیم که باید pop کنیم و مردد نیستیم.
ww^R ← وسط رشته قابل تشخیص نبوده و در هر مرحله pop را انجام می‌دهیم. پس غیرقطعی است.

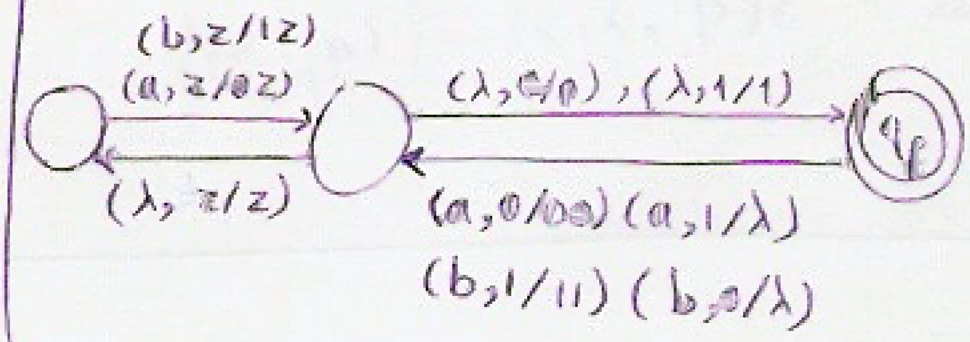
مثال) نشان دهید که $a^n b^{2n}$ الف) مستقل از متن معین است.



ب) معین است $L=\{a^n b^n \;\; n \ge 0\} \cup \{a\}$



ج) معین $L=\{w\in\{a,b\}^* : n_a(w)\ne n_b(w)\}$



گرامرهای LL (این نکات حفظ شود)

- هر گرامر LL یک گرامر غیرمبهم است.
- اگر G یک گرامر LL باشد، آنگاه L(G) یک زبان مستقل از متن معین است.
- یک زبان مستقل از متن معین هیچگاه ذاتاً گنگ نیست.

فصل ۸ - خواص زبانهای مستقل از متن

لم تزریق ابزار مناسبی برای نشان دادن اینکه برخی زبان ها منظم نیستند بود - برای زبان های CF هم لم داریم.

لم تزریق برای زبان های مستقل از متن:

فرض کنید L یک زبان مستقل از متن نامتناهی باشد. آنوقت عدد صحیح m هست بطوریکه برای هر $w \in L$ با $|w| \geq m$ می تواند بصورت زیر تجزیه شود:

$w = uvxyz$ و $|vxy| \leq m$ و $|vy| \geq 1$

(بطوریکه) $uv^ixy^iz \in L$ به ازای تمام مقادیر $i = 0, 1, 2, \dots$ باشد.

خواص زبان های CF:

① خانواده زبان های مستقل از متن تحت اجتماع، الحاق و بستار ستاره ای بسته است.

$S \to SS_1 | \lambda$ $S \to S_1S_2$

S_1 و S_2 مستقل از متن
با این $S \to S_1|S_2$
زبان حاصل هم CF است.

② خانواده زبان های CF تحت اشتراک و متمم گیری بسته نیست.

③ اگر L_1 زبان مستقل از متن و L_2 زبان منظم باشد، اشتراک آنها مستقل از متن است. به این نوع اشتراک، اشتراک منظم می گویند.

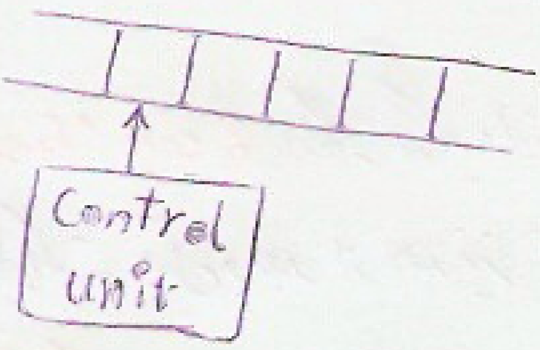
④ اگر یه زبان CF تهی باشد، از روی گرامرش قابل تشخیص است. (کافیه الگوریتم حذف دستورات اضافی را روی آن اجرا کنیم)

⑤ اگر یه زبان CF، نامتناهی باشد، با وجود حلقه در گرامرش قابل تشخیص است.

* صورت قضایایی که در آخر فصل هشتم خواسته که اثبات کنید را حفظ کنید (برای کنکور ارشد)

فصل ۹ - ماشین تورینگ

ماشین تورینگ استاندارد: یک آتاماتون است که حافظهٔ موقت آن یک نوار است که از هر دو طرف می‌تواند ادامه داشته باشد - اطلاعات روی نوار می‌تواند خوانده شود یا به هر ترتیبی تغییر کند (به سمت چپ و راست)



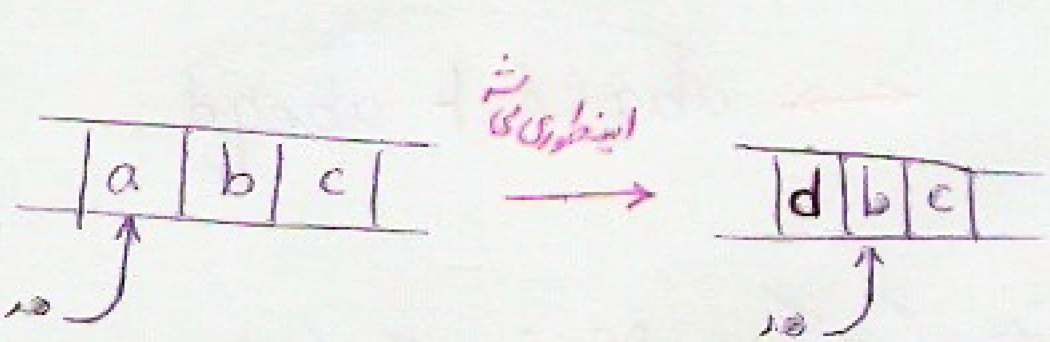
تعریف ماشین تورینگ: $M=(Q,\Sigma,\Gamma,\delta,q_0,\square,F)$

- Q: وضعیت‌ها
- Σ: الفبای ورودی
- Γ: الفبای نوار
- δ: تابع تغییر وضعیت
- q_0: شروع
- $\square$: blank
- F: وضعیت‌های پایانی

← در ماشین تورینگ فرض می‌کنیم که $\Sigma \subseteq \Gamma-\square$ یعنی الفبای ورودی، زیرمجموعهٔ الفبای نوار منهای $\square$ (خالی) است.

مثال) یک تابع δ نحوهٔ عملکرد ماشین را در هر وضعیت مشخص می‌کند - مثل: $\delta(q_0,a)=(q_1,d,R)$

در این حالت که نوشتیم، اگر در q_0 باشیم و a را از ورودی بخوانیم، به جای a مقدار d را می‌نویسیم و یکی روی نوار به سمت راست (R) می‌رویم.



نکته) order ماشین تورینگ n^2 یا بیشتر باشد، اما dfa فقط n هستند. چون یکبار می‌خوانند. اما ماشین تورینگ می‌تونه چندین بار بره و برگرده

چند نمونه تابع دیگه:

$\delta(q_0,a)=(q_0,b,R)$ ← a را روی نوار به b تبدیل می‌کند سپس یک خانه به راست می‌رود.

$\delta(q_0,\square)=(q_1,\square,L)$ ← blank را تغییر نمی‌دهد و بعد یکی به چپ می‌رود.

خصائص ماشین تورینگ:

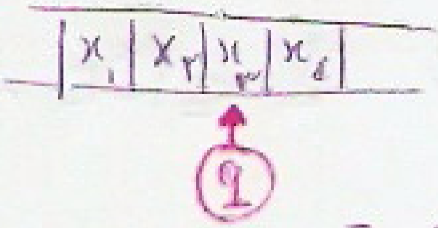
(۱) دارای نواری است که از چپ و راست نامحدود است.

(۲) ماشین تورینگ قطعی است – یعنی به ازای هر پیکربندی یک حرکت قابل تعریف است.

(۳) محتوای اولیهٔ روی نوار، مقادیر ورودی است – وقتی ماشین توقف کند، می‌توان بخشی از محتوای نوار را خروجی دانست.

پیکربندی یعنی: انگار در یک لحظه، عکسی از ماشین تورینگ بگیریم. در این عکس وضعیت q و نوک هد و نیز کاراکتر قبل و بعد از هد نیز قابل دید هستند.

نمایش پیکربندی: مثلاً بصورت $x_1x_2qx_3x_4$ نمایش داده می‌شه – q وضعیت کنونی است و نوک هد روی اولین خانهٔ بعد از q قرار دارد – اینجوری:



پس حالتی مثل $q\square w$ یعنی نوک هد روی $\square$ است.

حرکت: حرکت از یک پیکربندی به پیکربندی بعدی را با $\vdash$ نشان می‌دهند و برای چند حرکت از $\vdash^*$ استفاده می‌شود.

مثال) $\delta(q_1, c) = (q_2, e, R) \longleftrightarrow abq_1cd \vdash abeq_2d$

حلقهٔ بی‌نهایت) ماشینی که هرگز متوقف نشود، می‌گوییم که در حلقه بی‌نهایت افتاده – پس پذیرفته هم نمی‌شود – چون هرگز توقف نمی‌کند در حالت نهایی.

مثال ساده: روی $\Sigma=\{0,1\}$ ماشینی بنویسید که 00^* را قبول کند.

$$\delta(q_0, 0) = (q_0, 0, R)$$
$$\delta(q_0, \square) = (q_1, \square, R)$$

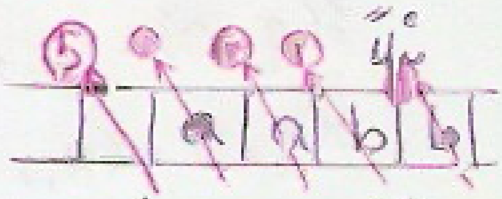
تا وقتی 0 ببینیم به راست می‌رویم. اگر 1 مشاهده کند در q_0 متوقف میشه – چون تعریف نشده و چون در یک وضعیت غیرنهایی توقف کردیم، رشته پذیرفته نمی‌شه – در اینجا اگه رشته تهی باشه هم پذیرفته می‌شود که به معنای پذیرش λ است.

ماشین تورینگ

پذیرنده : فقط یک رشته را Accept می‌کند.

تراگذر (کامپیوتر) : یه چیزی را محاسبه می‌کند.

تمرین) ماشین تورینگی طراحی کنید که $\{a^n b^n \mid n \geq 1\}$ را بپذیرد.



هر a را که دید، ماشین x می‌ذاره و میره اولین b را پیدا کرده و ماشین y می‌ذاره و همین ترتیب برای a بعدی.

$\delta(q_0, a) = (q_1, x, R)$

$\delta(q_1, a) = (q_1, a, R)$

$\delta(q_1, y) = (q_1, y, R)$

$\delta(q_1, b) = (q_2, y, L)$

$\delta(q_2, y) = (q_2, y, L)$

$\delta(q_2, a) = (q_2, a, L)$

$\delta(q_2, x) = (q_0, x, R)$

$\delta(q_0, y) = (q_3, y, R)$

$\delta(q_3, y) = (q_3, y, R)$

$\delta(q_3, \square) = (q_f, \square, R)$

ماشین تورینگ تراگذر (Computer) : هر چیز جز فضای خالی روی نوار، ورودی آن است - خروجی نیز چیزی است که بعد محاسبه روی نوار باقی می‌ماند.

تعریف) تابع f با دامنه D را محاسبه‌پذیر تورینگ گوییم هرگاه یک تورینگ برای هر رشتهٔ $w \in D$ به یک وضعیت نهایی برود.

مثال) ماشین تورینگ برای محاسبهٔ $x+y$ بسازید.

هر عدد را با تعدادی 1 روی نوار نشان میدیم. مثلاً ۵ به صورت 11111 خواهد بود.

جمع دو عدد را به صورت 111110111 نشان می دهیم. بعد برای جمع کافیه 0 را به آخر سمت راست ببریم.

$\delta(q_0,1)=(q_0,1,R)$

$\delta(q_0,0)=(q_1,1,R)$

$\delta(q_1,1)=(q_1,1,R)$

$\delta(q_1,\square)=(q_2,\square,L)$

$\delta(q_2,1)=(q_3,0,L)$

$\delta(q_3,1)=(q_3,1,L)$

$\delta(q_3,\square)=(q_f,\square,R)$

مثال) ماشین تورینگی که $L(a(a+b)^*)$ را قبول کند:

$\delta(q_0,a)=(q_1,a,R)$

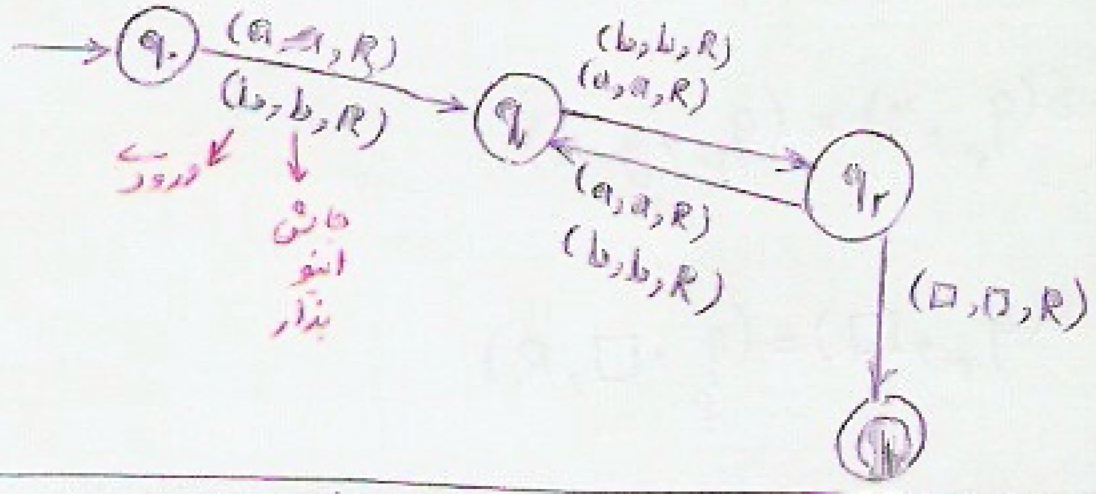
$\delta(q_1,a)=(q_1,a,R)$

$\delta(q_1,b)=(q_1,b,R)$

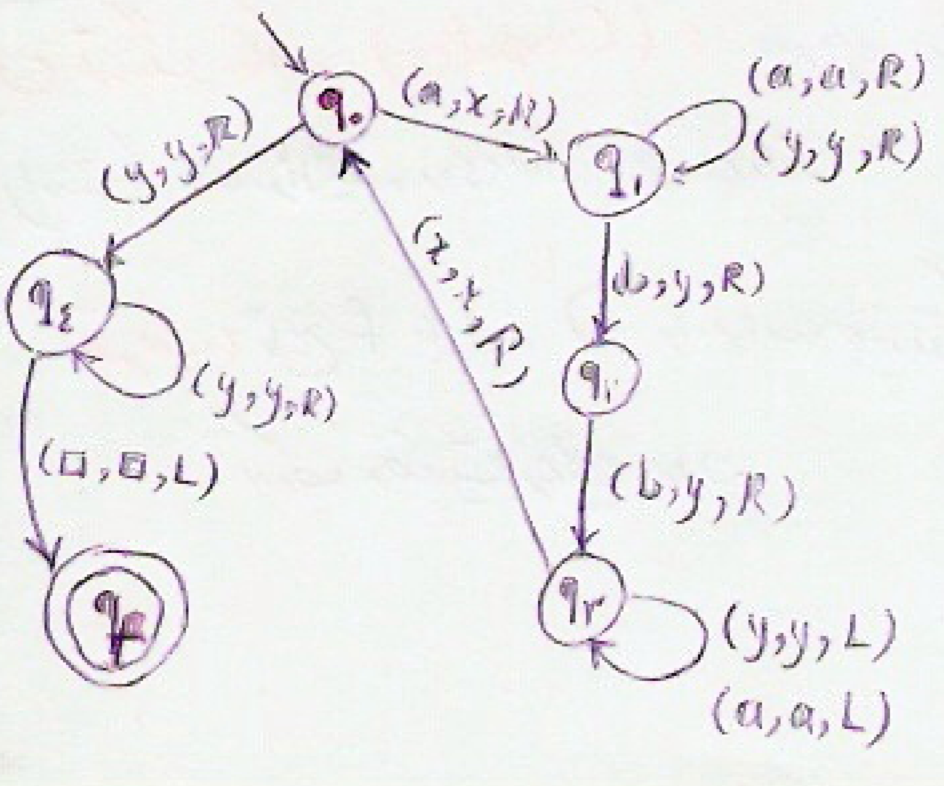
$\delta(q_1,\square)=(q_f,\square,L)$

مثال) ماشینی که $L=\{w : |w|\ \text{زوج است}\}$ را قبول کند.

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



مثال) تورینگ $L=\{a^n b^{2n} \quad n\geq 1\}$



بسمعه تعالی

سوالات میان ترم درس نظریه زبانها و ماشینها (گروه اول) استاد: نورالله

هر جواب درست منجر به ۱ نمره مثبت و هر جواب نادرست منجر به ۰/۵ نمره منفی خواهد شد.

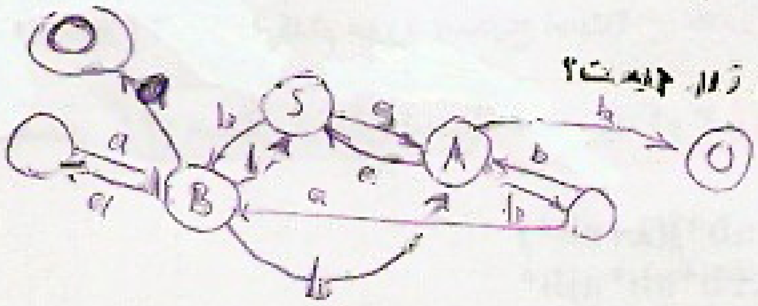
زمان: ۵۰ دقیقه

نام و نام خانوادگی:

۱. زبانی است شامل تمام رشته های a و b که دارای تعدادی فرد از کاراکتر b می باشد. Regular expression مربوط به این زبان کدام است؟

1) ba*(a|ba*b)* ×
۲) ab*(ab*ab*)* ×
۳) a*b(a*ba*b)* ×
۴) (a|ba*b)*.ba* صحیح

۲. زبان گرامر زیر چیست؟



S→aA|bB
A→bbA| baB | aS | b
B→ abA | aaB | a | bS

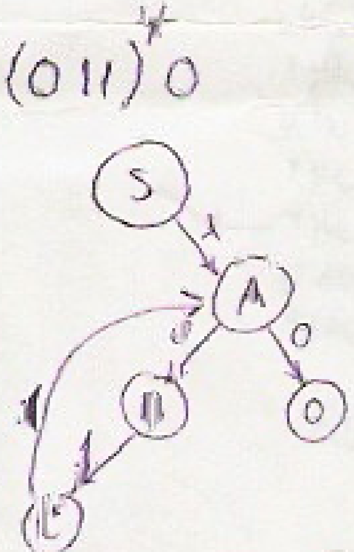
۱) فقط رشته هایی را تولید می کند که در آن ها تعداد a ها و b ها برابر هستند. مانند aababb
۲) فقط رشته هایی را تولید می کند که در آن ها تعداد زوجی a و تعداد زوجی b وجود دارد. مانند aabbaa
۳) فقط رشته هایی را تولید می کند که از تعداد فردی a و تعداد فردی b تشکیل شده اند. مانند aabab صحیح
۴) فقط رشته هایی را تولید می کند که در آن ها تعداد a ها از تعداد b ها بیش تر است. مانند babaa

۳. با فرض این که S علامت شروع گرامر زیر و λ نمایانگر رشته تهی باشد، در زبان گرامر چند رشته وجود دارد که دقیقا دارای سه پرانتز باز و حداکثر یک << و>> است؟

S→P
P→(E)
E→[illegible]
F→a
F→(P)
H→,SH
H→λ
[illegible]→(P)

۱) ۲
۲) ۳
۳) ۴
۴) ۱

(011)*0

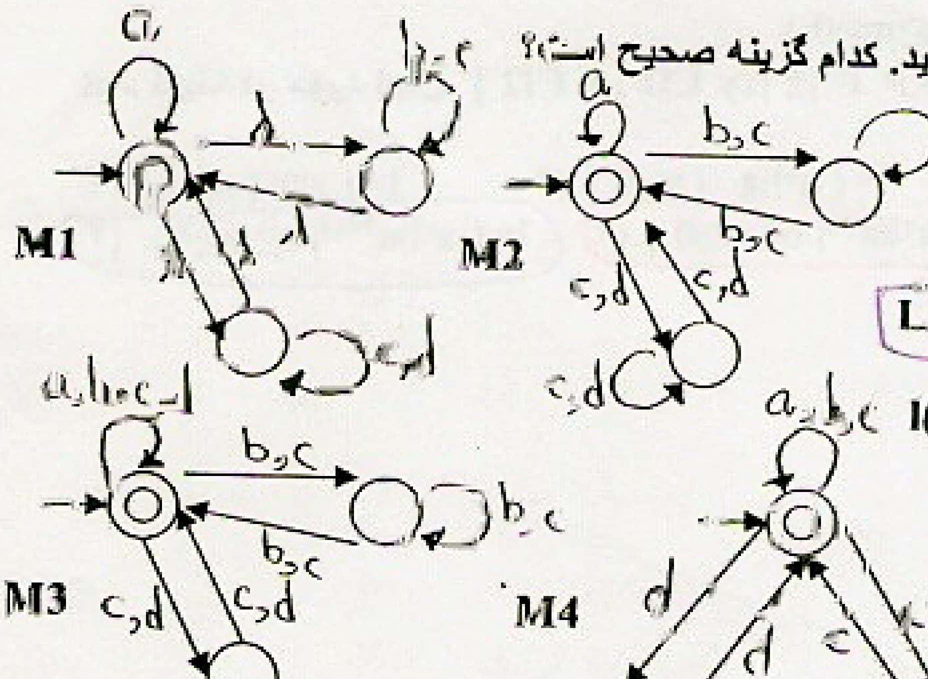


۴. گرامر

G: S→A
A→0
A→1B
B→1C
C→1A

۱) مبهم است
۲) نا مبهم است
۳) زبانی متناهی را تولید می کند.
۴) هیچ کدام

۵. اتومات های متناهی زیر را در نظر بگیرید. کدام گزینه صحیح است؟



۱) L(M4)⊂L(M2) و L(M1)∩L(M3)=∅
۲) L(M1)=L(M3) و L(M4)⊂L(M3)
۳) L(M1)⊂L(M3) و L(M2)⊂L(M4)
۴) L(M1)⊂L(M2) و L(M2)=L(M3)

۶. اگر L4 و L5 دو زبان منظم دلخواه باشند و زبانهای زیر نیز مفروض باشند کدام گزینه درست است؟

منظم $L1=\{ WW^RV \mid V,W \in\{a,b\}^* \}$

نامنظم. [illegible] $L2=\{ WcV \mid V,W \in\{a,b\}^* ,\ W\neq V\}$

منظم $L3=\{ W \mid W \in L4 ,\ W^R \in L5 \}$

۱) L3,L2,L1 نا منظم اند. ۲) L3,L2,L1 هر سه منظم اند.

۳) L3,L1 منظم ولی L2 نا منظم است. ۴) L2,L1 نامنظم ولی L3 منظم است.

$a^* b^*$ زبان [illegible] ۷) در مورد زبان $\{a^mb^n \mid m=n\} \cup \{a^nb^m \mid m\neq n\}$ کدام مورد صحیح است؟ هر زبان منتظم، مستقل از متن هم هست

۱) منظم است. ۲) مستقل از متن نیست. ۳) مستقل از متن است. ۴) مورد ۱ و ۳ bb aaa

۸. فرض کنید:

$R1=b^*(\lambda+ab^*ab^*)(\lambda+ab^*)$

$R2=(\lambda+b^*a)(\lambda+b^*ab^*a)b^*$

$R3=b^*(ab^*(\lambda+ab^*a)+b^*ab^*ab^*a))$

کدام یک از گزینه های زیر صحیح است؟

۱) $R3=R1, R1\neq R2$ ۲) $R3\neq R2 , R1=R2$ درسته

۳) $R3=R2 , R1=R2$ ۴) $R3=R2, R1\neq R2$

۹. گرامر:

$G: S\rightarrow AB$

$A\rightarrow BB|a$

$B\rightarrow BA|b$

را در نظر بگیرید. فرض کنید $w\in\{a,b\}^*$ و $|w|=n$

۱) اگر w توسط G تولید شود در حالت کلی الگوریتمی با پیچیدگی زمانی $o(n^2)$ وجود دارد که درخت اشتقاق برای w را بدست آورد.

۲) در حالت کلی الگوریتمی وجود ندارد که بتوان گفت آیا w توسط G تولید می شود یا خیر.

۳) در حالت کلی الگوریتمی با پیچیدگی زمانی $o(n^2)$ وجود دارد که بتوان گفت آیا w توسط G تولید می شود یا خیر.

۴) هیچ کدام

۱۰. زبانهای مستقل از متن l1,l2 به صورت زیر مفروضند:

$l1 = \{$[illegible]$\}$ $l2=\{ a^nb^m \mid n\geq m\geq 0 \}$

کدام گزینه در مورد زبان $l=\{x \mid xy \in l1 ,\ y \in l2 \}$ درست است؟

۱) $l=\{$[illegible]$\mid n\geq 0\}$ ۲) $l=\{ a^nba \mid n\geq 0 \}$

۳) $l=\{$[illegible]$\}$ ۴) $l=\{ a^nba^{m+1} \mid n\geq m\geq 0 \}$

1/

تمرین سری دوم نظریه زبان ([illegible])

① برای هر یک از زبان های زیر dfa رسم کنید:

الف $L = \{ w \mid w \in \{0 \cup 1 \cup 2 \cup 3\}^*$ و w مضرب ۶ است $\}$ حتماً حل شود:

ب $L = (a \cup b)^* - a^* b^*$ $\quad \Sigma = \{a, b\}$

ج $L = (a^* b^+ c^* \cup c^+ a^* b^+)^*$ $\quad \Sigma = \{a, b, c\}$

د $L = ((abab)^* + (aaa^* + b)^*)$

ه $L = \{ w : (n_a(w) - n_b(w)) \bmod 3 > 0 \}$ حتماً حل شود

و $L =$ تمام رشته هایی که حداقل a2 و حداکثر b3 دارند. حتماً حل شود

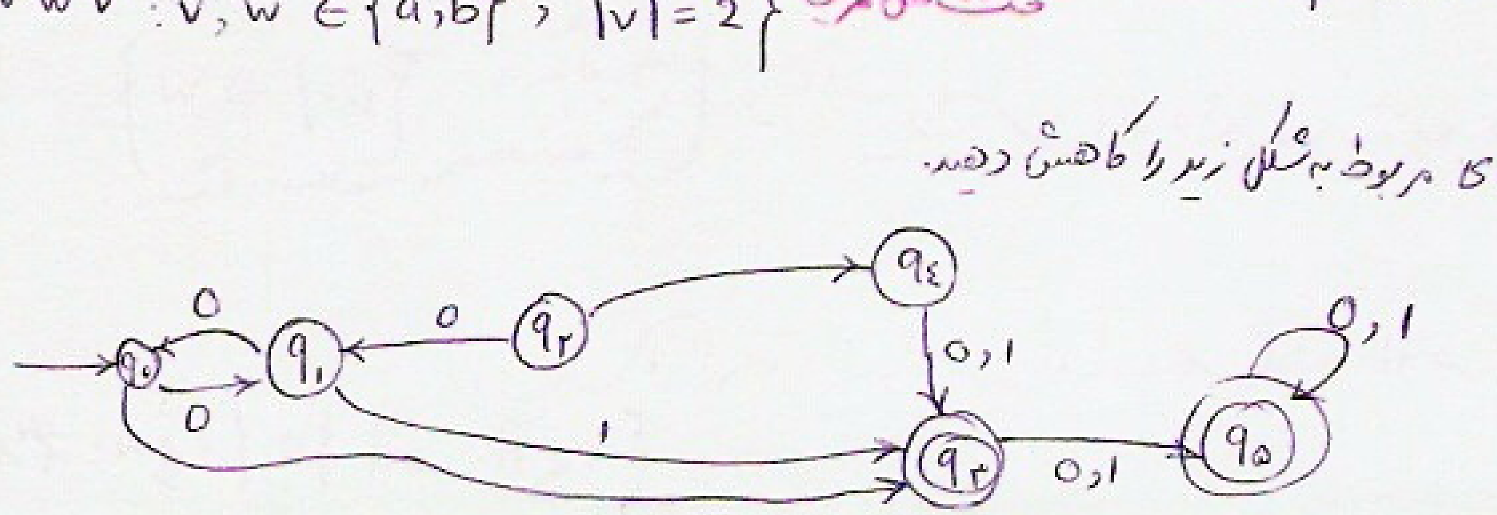
ی $L = \{ w :$ حداکثر دوتا دنباله از a ها بطول ۳ وجود داشته باشد $\}$ حتماً حل شود

ز) $L = \{ w : n_a(w) \bmod 3 > n_b(w) \bmod 3 \}$

② یک ماشین DFA حداکثر با سه حالت طراحی کنید که کلیه رشته های اعدادی که از 0 تا 9 $(\Sigma = \{0, 1, 2, \dots, 9\})$ تشکیل شده و آن دسته از رشته های عددی که بر سه بخش پذیر نیست را بپذیرد.

③ نشان دهید که زبان روبرو منظم است: $L = \{ vwv : v, w \in \{a, b\}^*, |v| = 2 \}$ صفحه ۵۲ حل تمرین

④ وضعیت های آتاماتای مربوط به شکل زیر را کاهش دهید.



تمرینات سری سوم نظریه زبان

۱) کدام یک از زبان های زیر، مستقل از متن قطعی است؟ با علت بگویید:

$\{a^n b^m : n<m\}$

$\{a^n w w^R a^n \mid w \in a^*, n>0\}$

$\{a^n b^n :$ n عددی فرد است $\}$

$\{a^+ w w^R a^+ \mid w \in (a,b)^*\}$

$\{a^n b^m : n>m\}$

---

۲) npda رسم کنید.

(۳ نمره) $2\, n_a(w) \leq n_b(w) \leq 3\, n_a(w)$

(۲ نمره) $n_a(w) = 2\, n_b(w)$

(۲/۵ نمره) $\{a^n b^m c^k : n+2m = k\}$

(۱/۵ نمره)

۳) اینو به فرم گریباخ در بیارین:

$S \rightarrow aAB \mid AB \mid SS \mid aSS$

$A \rightarrow AB \mid Aa$

$B \rightarrow bB \mid b$

این غلطه. چون A هرگز به ترمینال ختم نمی‌شه

---

۴) به فرم چامسکی درآورید:

$S \rightarrow abAB$

$A \rightarrow bAB \mid \lambda$

$B \rightarrow BAa \mid A \mid \lambda$

تمرین سری چهارم نظریه زبان

۱) صفحه ۲۴۷ تست ۵

۲) برای این زبان‌ها ماشین تورینگ طرح کنید:

الف) $n_a(w) = 2n_b(w)$

ب) $2n_a(w) \leq n_b(w) \leq 3n_b(w)$

ج) $a^n b^{2n}$

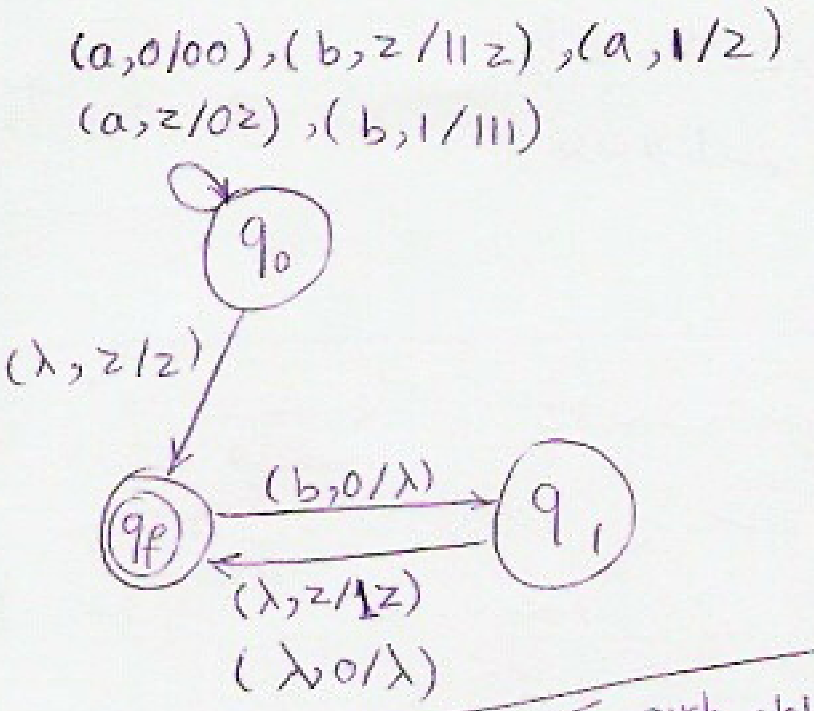
د) $a^n b^n a^n b^n$

هـ) $n_a(w) = n_b(w) + 1$

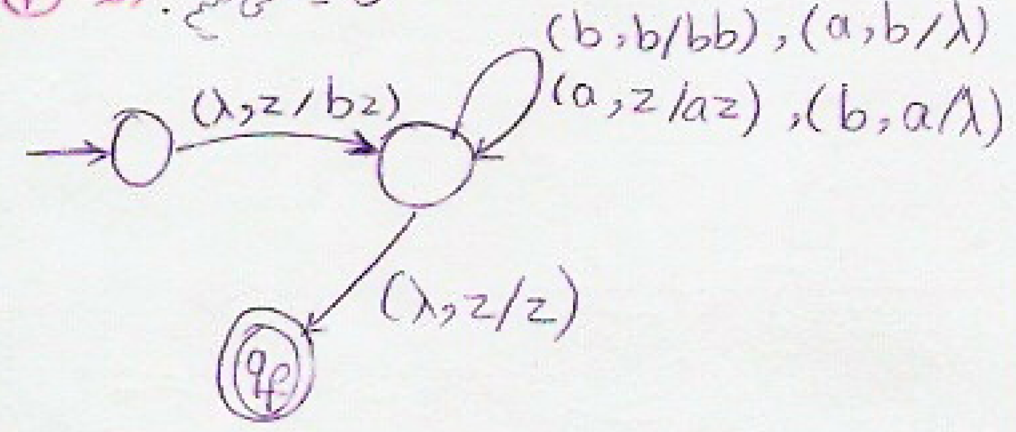
۳) npda رسم کنید (ماشین پشته‌ای نامعین)

$$\{w_1 c w_2 : w_1, w_2 \in \{a+b\}^*, w_1 \neq w_2^R\}$$

۲) الف



۲) هـ. یه دونه b اول push می‌کنیم، بعد مساوی مقایسه می‌کنیم.



۴) اینو به فرم گریباخ درآورید:

$S \rightarrow aAB \mid AB \mid SS \mid aSS$

$A \rightarrow AB \mid Aa \mid b$

$B \rightarrow bB \mid b$

صفحه ۱۸۳ تمرین ۲۱ را برای تبدیل SS در این نمونه گریباخ‌ها لازم داریم.
(بازگشتی از چپ)

تمرین سری ۵ نظریه

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

نشان) کدام وابسته به متن بنویسید.

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

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

$L = \{a^n b^m c^k : 0 < n < m < k\}$

ماشین تورینگ برای زبان روبرو بکشید:

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

a	a	b	c	c

$\delta(q_0, a) = (q_1, x, R)$

$\delta(q_1, a) = (q_1, a, R)$

$\delta(q_1, b) = (q_2, b, R)$

$\delta(q_2, b) = (q_2, b, R)$

$\delta(q_2, c) = (q_4, y, L)$

$\delta(q_2, y) = (q_3, y, R)$

$\delta(q_3, y) = (q_3, y, R)$

$\delta(q_3, c) = (q_4, y, L)$

$\delta(q_4, y) = (q_4, y, L)$

$\delta(q_4, b) = (q_4, b, L)$

$\delta(q_4, a) = (q_4, a, L)$

$\delta(q_4, x) = (q_0, x, R)$

$\delta(q_0, b) = (q_5, b, R)$

$\delta(q_5, b) = (q_5, b, R)$

$\delta(q_5, y) = (q_5, y, R)$

$(q_5, \square) = (q_f, \square, L)$

ماشین تورینگی که تعداد a و b مساوی را قبول کند:

برای این کار اولین کاراکتری که دیدیم علامت زده و x می‌کنیم و می‌رویم جلو تا به کاراکتر معکوسش برسیم

$\delta(q_0, a) = (q_a, x, R)$

$\delta(q_0, b) = (q_b, x, R)$

$\delta(q_0, x) = (q_0, x, R)$

$\delta(q_0, \square) = (q_f, \square, L)$

$\delta(q_a, x) = (q_a, x, R)$

$\delta(q_b, x) = (q_b, x, R)$

$\delta(q_a, a) = (q_a, a, R)$

$\delta(q_b, a) = (q_b, b, R)$

$\delta(q_a, b) = (q_r, x, L)$

$\delta(q_b, a) = (q_r, x, L)$

$\delta(q_r, a) = (q_r, a, L)$

$\delta(q_r, b) = (q_r, b, L)$

$\delta(q_r, x) = (q_r, x, L)$

$\delta(q_r, \square) = (q_0, \square, R)$

چند سوال از درس نظریه:

① زبان گرامر زیر چیست؟

گرامر:

$S \to \lambda | AB$

$A \to S | 1A$

$B \to S | 0B$

الف) $(1^*0^*1^*)^+$: هر ترکیب از 0 و 1 را می‌دهد.

ب) $(1^*0^*)^* \cup \{\lambda\}$: مثل بالایی است.

ج) $(1^+0+01^+)^*$

د) $(1^+0^+0^+1^+) \cup \{\lambda\}$

این دوتا ۳ تا صفر پشت هم را نمی‌دهد.

گزینه الف و ب صحیح است.

---

② توسط npda زیر چی تولید میشه؟ (داخل q_2 پذیرفته شده است)

$\delta(q_1, a, z) = \{(q_1, a), (q_2, \lambda)\}$ a تک قابل تولیده

$\delta(q_1, b, a) = \{(q_1, b)\}$

$\delta(q_1, b, b) = \{(q_1, b)\}$

$\delta(q_1, a, b) = \{(q_2, \lambda)\}$

الف) $\{a^n ba : n \geq 1\} \cup \{b\}$ غلط

ب) $\{ab^n a : n \geq 1\}$: a تک را تولید نمی‌کند.

ج) $\{ab^n a : n \geq 1\} \cup \{a\}$: درست.

د) $\{ab^n a : n \geq 0\}$: a تک را تولید نمی‌کنه.

---

③ order تورینگ ماشینی که $a^n b^n$ را تولید کند، چیه؟

$O(n^2)$

---

④ فرض کنید L_1 و L_2 منظم هستند. کدام منظم است؟

الف) $L_3 = \{w \mid w \in L_1, w^R \in L_2\}$

ب) $L_4 = \{w \mid w \in L_1, w \in L_2, |w| = 2n+1, n \geq 0\}$

ج) $L_5 = \{w \mid w \notin L_1, w \notin L_2\}$

دلیل $w \in \bar{L_1} \cap \bar{L_2}$

د) هر ۳ زبان منظم هستند.

---

⑤ ماشینی که فقط بر روی نوار به سمت راست حرکت می‌کند، زبان منظم دارد.

چون استک ندارد.

" از نوار استفاده نمی‌کنه.

فقط تغییر حالت می‌دهد.

---

⑥ این گرامر چی تولید می‌کنه؟

$S \to S_1 B$

$S_1 \to aS_1 b$

$bB \to bbbB$

$aS_1 b \to aa$

$B \to \lambda$

یعنی این خط $a^n b^n$ تولید می‌کند.

این خط ② تا aa به رشته $a^n b^n$ اضافه می‌کند. یعنی $a^{n+2} b^n$ یا $(a^{n+1} b^{n-1})$

اینجا هر B دوتا bb تولید می‌کند (b^{2k})

اینجا هم که B تموم میشه ← در نهایت $a^{n+1} b^{n-1} b^{2k}$ تولید می‌شود.

مکمل یک زبان بازگشتی شمارا، بازگشتی است: (غلط)

" " ، بازگشتی شمارا است: (درست)