مبانی نظریه محاسبه

معرفی درس

منابع

منبع اصلی:

- Linz, D.: An Introduction to Formal Languages and Automata, 3rd Edition; Jones and Barlett Publishers 2001

- ترجمه: مقدمه‌ای بر نظریه زبان‌ها و ماشین‌ها

فصل اول

مقدمه‌ای بر نظریه محاسبات

مجموعه

تعریف: مجموعه

گردایه‌ای از عناصر غیر تکراری که ترتیب بین آنها اهمیت ندارد.

برای اینکه نشان دهیم x عضوی از S است، می‌نویسیم: $x \in S$؛ خلاف این گزاره را با $x \notin S$ نشان می‌دهیم.

برای نمایش عناصر یک مجموعه آنها را داخل آکولاد قرار می‌دهیم. نمونه

$$S = \{۰, ۱, ۲\}.$$

اعمال روی مجموعه‌ها

سه عمل **اجتماع** ($\cup$)، **اشتراک** ($\cap$)، و **تفاضل** ($-$) روی مجموعه‌ها، به صورت زیر تعریف شده است:

$$S_1 \cup S_2 = \{x : x \in S_1 \text{ or } x \in S_2\}$$

$$S_1 \cap S_2 = \{x : x \in S_1 \text{ and } x \in S_2\}$$

$$S_1 - S_2 = \{x : x \in S_1 \text{ and } x \notin S_2\}$$

تعریف: مجموعه جهانی (U)

مجموعه‌ای شامل تمامی عناصر ممکنه.

- متمم مجموعه S با $\overline{S}$ نشان داده می‌شود که شامل تمامی عناصری است که در S وجود ندارند. با زبان ریاضیات می‌توان گفت:

$$\overline{S} = \{x : x \in U, x \notin S\}$$

- مجموعه‌ای که هیچ عضوی ندارد **مجموعه تهی** یا **مجموعه پوچ** نامیده می‌شود و با $\emptyset$ نشان داده می‌شود.

واضح است که:

$$S \cup \emptyset = \emptyset \cup S = S$$

$$S \cap \emptyset = \emptyset$$

$$\overline{\emptyset} = U$$

$$\overline{\overline{S}} = S$$

قوانین دمورگان

$$\overline{S_۱ \cup S_۲} = \overline{S}_۱ \cap \overline{S}_۲$$

$$\overline{S_۱ \cap S_۲} = \overline{S}_۱ \cup \overline{S}_۲$$

روابط بین دو مجموعه

تعریف: زیر مجموعه

مجموعه S_1 زیر مجموعه S است، $S_1 \subseteq S$، اگر تمامی عناصر S_1 در S باشند.

تعریف: زیر مجموعه محض

اگر S عنصری داشته باشد و S_1 آن را نداشته باشد، آنگاه می‌گوییم S_1 یک **زیر مجموعه محض** از S است و می‌نویسیم: $S_1 \subset S$

تعریف: مجموعه‌های مجزا

اگر S_1 و S_2 عنصر مشترکی نداشته باشند، یعنی، $S_1 \cap S_2 = \emptyset$، آن دو را **مجموعه‌های مجزا** می‌نامیم.

تعریف: مجموعه‌ای که تعداد محدودی عنصر داشته باشد **مجموعه محدود یا متناهی** و در غیر این صورت **نامحدود یا نامتناهی** نامیده می‌شود.

سه مفهوم اساسی نظریه محاسبات: زبان، گرامر، و آتاماتا

تعریف: الفبا

به سادگی، یک مجموعه غیر تهی از نشانه‌ها است.

تعریف: رشته

دنباله‌ای متناهی از نشانه‌های الفبا را یک **رشته** می‌گوییم.

مثال: اگر الفبا $\Sigma = \{a, b\}$ باشد. آنگاه $abab$، $bbba$ و aaa رشته‌های از الفبای Σ هستند.

- در این درس از حروف کوچک آغازین الفبای انگلیسی (a، b، c، ...) برای نشان دادن عناصر الفبا، و از حروف کوچک پایانی الفبای انگلیسی (u، v، w، ...) برای نشان دادن رشته‌ها استفاده می‌کنیم.

مثال:

$$w = abaaa$$

نشان می‌دهد که رشته w مقدار $abaaa$ را دارد.

تعریف: اتصال

اگر

$$w = a_1 a_2 \cdots a_n,$$

$$v = b_1 b_2 \cdots b_m$$

باشد، آنگاه اتصال این دو رشته را با wv نشان می‌دهیم و داریم:

$$wv = a_۱a_۲ \cdots a_n b_۱ b_۲ \cdots b_m$$

تعریف: معکوس

اگر

$$w = a_۱a_۲ \cdots a_n$$

باشد، آنگاه معکوس آن، w^R اینگونه تعریف می‌شود:

$$w^R = a_n a_{n-۱} \cdots a_۱$$

تعریف: طول رشته

طول رشته w به صورت $|w|$ نشان داده می‌شود و تعداد حروف آن را بیان می‌کند.

تعریف: رشته تهی

رشته‌ای که طول آن صفر باشد، یعنی، هیچ حرفی نداشته باشد؛ رشته تهی را با λ نشان می‌دهیم.

بدیهی است که:

$$|\lambda| = ۰,$$

$$\lambda w = w\lambda = w$$

تعریف: زیر رشته، پیشوند، و پسوند

یک زیر رشته بخشی از حروف پشت سر هم یک رشته است. اگر $w = vu$ باشد؛ زیر رشته v را پیشوند و زیر رشته u را پسوند می‌گوییم.

مثال: اگر $w = abbab$ باشد، آنگاه $\{\lambda, a, ab, abb, abba, abbab\}$ مجموعه پیشوندهای w هستند و bab یکی از پسوندهای آن است.

اگر u و v دو رشته باشند طول اتصال آنها مجموع طول‌های مجزای دو رشته می‌باشد:

$$|uv| = |u| + |v| \qquad (۱-۱)$$

اثبات: معادله ۱–۱

با کمک استقرا روی طول رشته v این معادله را ثابت می‌کنیم. اگر $v = a$ آنگاه

$$|a| = ۱,$$

$$|wa| = |w| + ۱$$

برای هر $a \in \Sigma$ و هر رشته w برقرار است. زیرا، طول هر نشانه الفبا یک و با افزودن یک نشانه به یک رشته طول آن یک واحد افزایش می‌یابد.

- با توجه به تعریف معادله ۱–۱ برای هر u و هر v به طول یک برقرار است. این گزاره پایه استقراء می‌باشد.

- فرض استقراء این است که معادله ۱−۱ برای هر u و هر v با طول ۱، ۲، ۰۰۰، n برقرار است.
- حال باید برای v با طول $n+۱$ رابطه را اثبات کنیم.

رشته v را می‌توان اینگونه بازنویسی کرد: $v = wa$. که در آن $|w| = n$ است. بنابراین:

$$|v| = |w| + ۱,$$

$$|uv| = |uwa| = |uw| + ۱$$

اما با توجه به فرض استقرا داریم

$$|uw| = |u| + |w|,$$

$$|uv| = |u| + |w| + ۱ = |u| + |v|$$

بنابراین معادله ۱−۱ برای u و v با هر اندازه‌ای برقرار است. □

- تکرار رشته w به تعداد n بار را با w^n نشان می‌دهیم؛ برای مورد ویژه $n = 0$ داریم: $w^0 = \lambda$

- اگر Σ یک الفبا باشد، Σ^* مجموعه تمامی رشته‌هایی است که با اتصال صفر یا بیشتر از حروف Σ بدست می‌آید. این مجموعه همواره شامل λ است. اگر بخواهیم رشته تهی در آن نباشد از Σ^+ استفاده می‌کنیم: $\Sigma^+ = \Sigma^* - \lambda$.

تعریف: زبان

یک زبان زیر مجموعه‌ای از Σ^* است.

مثال: اگر $\Sigma = \{a, b\}$ باشد، آنگاه

$$\Sigma^* = \{\lambda, a, b, aa, ab, ba, bb, aaa, aab, \cdots\}$$

- مجموعه $\{a, aa, aab\}$ یک زبان متناهی روی الفبای Σ است.
- مجموعه

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

یک زبان (نامتناهی) روی الفبای Σ است؛ رشته‌های ab و $aabb$ در این زبان وجود دارند، اما، رشته aab به این زبان تعلق ندارد.

از آنجا که زبان‌ها مجموعه هستند بنابراین اعمال اجتماع، اشتراک، و تفاضل

زبان قابل تعریف است.

تعریف: متمم یک زبان $\overline{L}$

با توجه به مجموعه مرجع قابل تعریف است: $\overline{L} = \Sigma^* - L$

تعریف: عکس یک زبان L^R

مجموعه‌ای است که تمام رشته‌های آن معکوس شده است:

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

تعریف: اتصال دو زبان $L_1 L_2$

اتصال دو زبان را می‌توان اینگونه ساخت: $L_1 L_2 = \{xy : x \in L_1, y \in L_2\}$.

L^n به معنای اتصال n بار L در خودش است. برای حالت ویژه $n = ۰$ داریم $L^۰ = \lambda$. همچنین:

$$L^* = L^۰ \cup L^۱ \cup \cdots ,$$

$$L^+ = L^۱ \cup L^۲ \cup \cdots .$$

گرامر

فصل دوم

آتاماتای متناهی

آتاماتا، ماشینی ساده برای انجام محاسبات است. هدف از مطالعه آتاماتاها در این فصل استفاده آنها برای برای تشخیص دادن اینکه یک رشته به یک زبان تعلق دارد یا نه است. البته آتاماتاها کاربردهای بسیار بیشتر دارند.

تعریف آتاماتا

تعریف: آتاماتا

آتاماتا یک مدل انتزاعی از کامپیوتر است با این ویژگی‌ها:

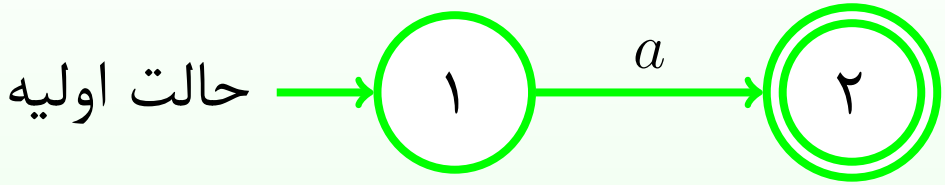
- مکانیزمی برای خواندن رشته ورودی دارد. این مکانیزم می‌تواند هر بار یک حرف از ورودی را خواندهو انتهای آن را تشخیص دهد.
- می‌تواند حافظه‌ای از اندازه بی‌نهایت داشته باشد و محتوای سلول‌های آن را تغییر دهد. الفبای حافظه موقت لزوماً همان الفبای ورودی نیست.
- یک واحد کنترل دارد که در هر لحظه در یکی از وضعیت‌های متناهی داخلی خود است.

- آتاماتا در هر زمان در یک وضعیت خاص است و حرف خاصی از ورودی را می‌خواند. وضعیت بعدی توسط **تابع تغییر وضعیت** تعیین می‌گردد. این تابع بر اساس علامت ورودی، وضعیت فعلی و محتوای حافظه وضعیت بعدی را معین می‌کند که می‌تواند باعث تولید خروجی و یا تغییر در حافظه شود.

- وضعیت و ورودی حافظه یک **پیکربندی** را می‌سازنند.

یک آتاماتا را می‌توان بوسیله یک گراف نشان داد که در آن رئوس حالت‌های داخلی و یال‌ها حالت تغییر وضعیت هستند. برچسب یال‌ها نشان می‌دهد که موقع گذر چه ورودی باید وجود داشته باشد.

یک اتوماتای ساده

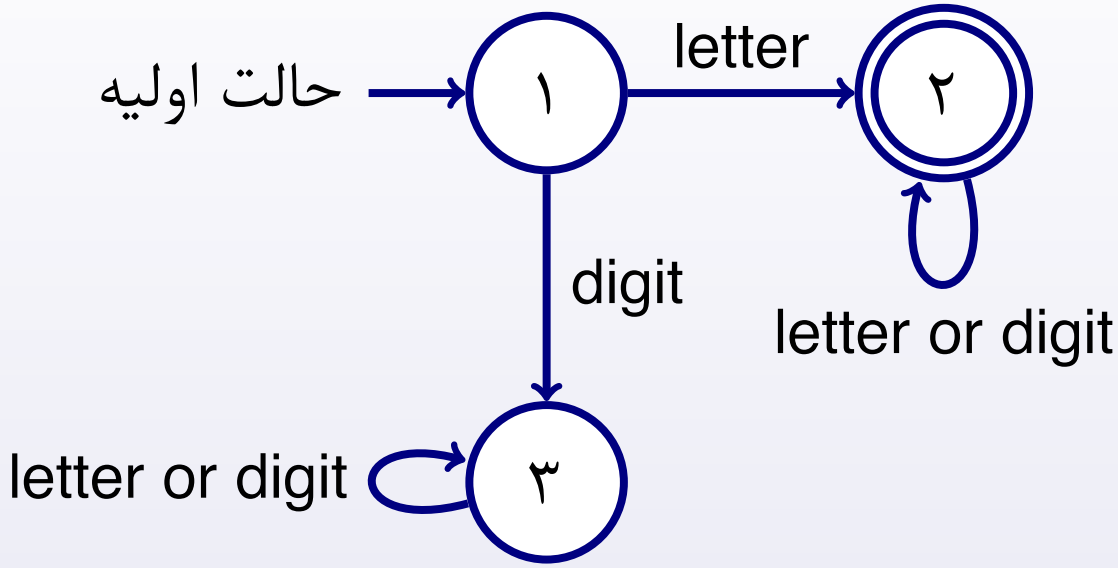


در این اتوماتا انتقال از ۱ به ۲ موقعی اتفاق می‌افتد که مکانیزم ورودی a را بخواند.

- حالت شروع آتاماتا را با یک پیکان ورودی نشان می‌دهیم

- حالات نهایی گره‌های هستند با دو دایره متداخل.
- رشته ورودی از چپ به راست، حرف به حرف خوانده می‌شود.

مثال: آتاماتای که همه نشانه‌های زبان پاسکال را می‌پذیرد.



اگر علامت اول ورودی یک حرف باشد به وضعیت ۲ که یک حالت پایانی است می‌رویم و با دیدن هر چیزی از آنجا بیرون نمی‌رویم. اگر علامت اول یک عدد باشد به وضعیت ۳ می‌رویم و همانجا می‌مانیم.

انواع آتاماتا: معین و نامعین

۱. **آتاماتای معین:** اگر وضعیت فعلی، ورودی و محتوای حافظه را بدانیم می‌توانیم رفتار بعدی آتاماتا را دقیقاً تعیین کنیم.

۲. **آتاماتای نامعین:** در این گونه با داشتن وضعیت فعلی، ورودی و محتوای حافظه چندین رفتار بعدی ممکن است.

در ادامه تعریف رسمی گونه‌ای ساده از آتاماتاها موسوم به **پذیرنده متناهی معین** آمده است.

تعریف: یک پذیرنده متناهی معین (DFA) یک پنج تایی مرتب مانند $M = (Q, \Sigma, \delta, q., F)$ است که در آن:

مجموعه‌ای متناهی از وضعیت‌ها است	Q
مجموعه‌ای از نشانه‌ها است که الفبای ورودی نامیده می‌شود	Σ
یک تابع کلی است به نام تابع تغییر حالت	$\delta : Q \times \Sigma \mapsto Q$
وضعیت اولیه است	$q.$
مجموعه وضعیت‌های نهایی است.	$F \subseteq Q$

عمل‌کرد پذیرنده متناهی معین

در زمان شروع فرض می‌شود که در وضعیت q_0 و مکانیسم ورودی روی اولین نشانه سمت چپ رشته ورودی می‌باشد. در هر حرکت آتاماتا، مکانیسم ورودی یک حرف به سمت راست می‌رود. بنابراین در هر حرکت یک نشانه ورودی مصرف می‌شود. وقتی به انتهای رشته رسیدیم،

1. اگر در یکی از وضعیت‌های پایانی باشیم رشته **پذیرفته** می‌شود،
2. چنانچه وضعیت پایانی نباشد رشته ورودی **رد** می‌شود.

گذر از یک وضعیت به وضعیت دیگر مطابق تابع گذر δ انجام می‌گیرد. برای نمونه

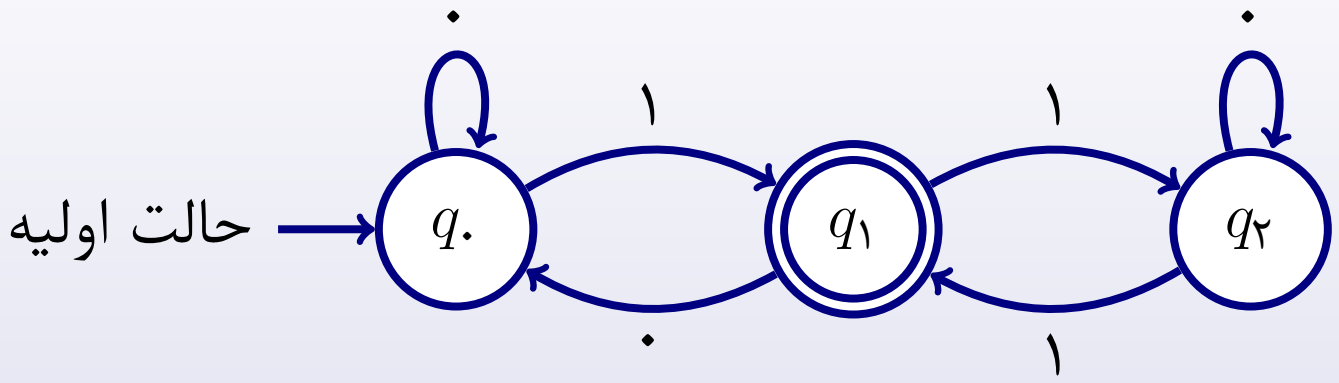
$$\delta(q_0, a) = q_1$$

یعنی، اگر DFA در وضعیت q_0 و نشانه ورودی a باشد، DFA به حالت q_1 می‌رود.

نمایش آتاماتا با کمک گراف‌ها

برای ساده‌تر شدن تصور کردن آتاماتاها، با کمک **گراف‌های گذار** آنها را رسم می‌کنیم. بدین صورت که رئوس نشان دهنده حالت‌ها و یال‌ها نشان دهنده تغییر وضعیت می‌باشند. برچسب رئوس نام حالت‌ها و برچسب یال‌ها سمبل ورودی جاری می‌باشد. برای مثال اگر q_0 و q_1 حالت‌های داخلی یک DFA باشد، آنگاه گراف مربوط به آن یک راس با برچسب q_0 و یک راس با برچسب q_1 دارد، بعلاوه، یال (q_0, q_1) برچسب a دارد اگر $\delta(q_0, a) = q_1$ باشد.

مثال:



گراف بالا این DFA را نشان می‌دهد:

$$M = (\{q_۰, q_۱, q_۲\}, \{۰, ۱\}, \delta, q_۰, \{q_۱\})$$

که δ اینگونه تعریف شده است:

$$\delta(q_۰, ۰) = q_۰, \delta(q_۰, ۱) = q_۱,$$

$$\delta(q_۱, ۰) = q_۰, \delta(q_۱, ۱) = q_۲,$$

$$\delta(q_۲, ۰) = q_۲, \delta(q_۲, ۱) = q_۱,$$

تعریف: گسترش تابع δ

گسترش تابع δ تابع $\delta^* : Q \times \Sigma^* \mapsto Q$ دومین ورودی آن یک رشته است و نشان می‌دهد که آتاماتا بعد از خواندن آن رشته در کدام حالت قرار خواهد گرفت. به صورت رسمی می‌توان δ^* را با بازگشت بدست آورد:

$$\delta^*(q, \lambda) = q,$$

$$\delta^*(q, wa) = \delta(\delta^*(q, w), a).$$

تعریف: زبان پذیرفته شده توسط یک DFA

زبان پذیرفته شده توسط $M = (Q, \Sigma, \delta, q., F)$ مجموعه تمامی رشته‌های روی الفبای Σ است که توسط M پذیرفته می‌شود. به عبارت رسمی‌تر:

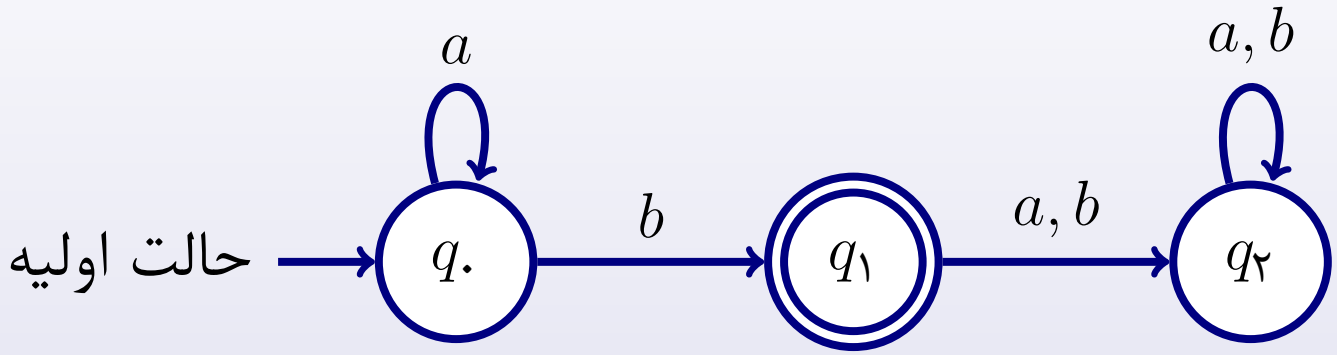
$$L(M) = \{w \in \Sigma^* : \delta^*(q., w) \in F\}$$

تعریف: مکمل یک DFA

مکمل یک DFA تمامی رشته‌های که DFA رد می‌کند را می‌پذیرد:

$$\overline{L(M)} = \{w \in \Sigma^* : \delta^*(q., w) \notin F\}$$

مثال: آتاماتای که زبان $L = \{a^n b : n \geq ۰\}$ را می‌پذیرد:



زبان‌های منظم

تعریف: زبان منظم

زبان L منظم نامیده می‌شود اگر و تنها اگر یک DFA مانند M باشد به طوری که

$$L = L(M).$$

- برای اثبات اینکه زبان L منظم بودن، باید DFA ی مانند M پیدا کنیم که آن را بپذیرد. یعنی، $L = L(M)$.

باشد

- به یک مجموعه از زبان‌ها یک **خانواده** نیز می‌گویند.
- خانواده زبان‌هایی که توسط DFA ها پذیرفته می‌شوند کاملاً محدود است.

پذیرنده متناهی نامعین (NFA)

- به سادگی، در پذیرنده متناهی نامعین (NFA) با حضور در یک وضعیت و رویت یک نشانه روی ورودی، بیش از یک کار برای انجام دادن وجود دارد.
- اگر اجازه داده شود تا پذیرنده متناهی به طور نامعین کار کند، پیچیدگی آن بسیار بالا می‌رود.

تعریف: یک پذیرنده متناهی نامعین (NFA) یک پنج تایی مرتب مانند $M = (Q, \Sigma, \delta, q., F)$ است که در آن:

مجموعه‌ای متناهی از وضعیت‌ها است	Q
مجموعه‌ای از نشانه‌ها است که الفبای ورودی نامیده می‌شود	Σ
یک تابع کلی است به نام تابع تغییر حالت	$\delta : Q \times (\Sigma \cup \{\lambda\}) \mapsto ۲^Q$
وضعیت اولیه است	$q.$
مجموعه وضعیت‌های نهایی است.	$F \subseteq Q$

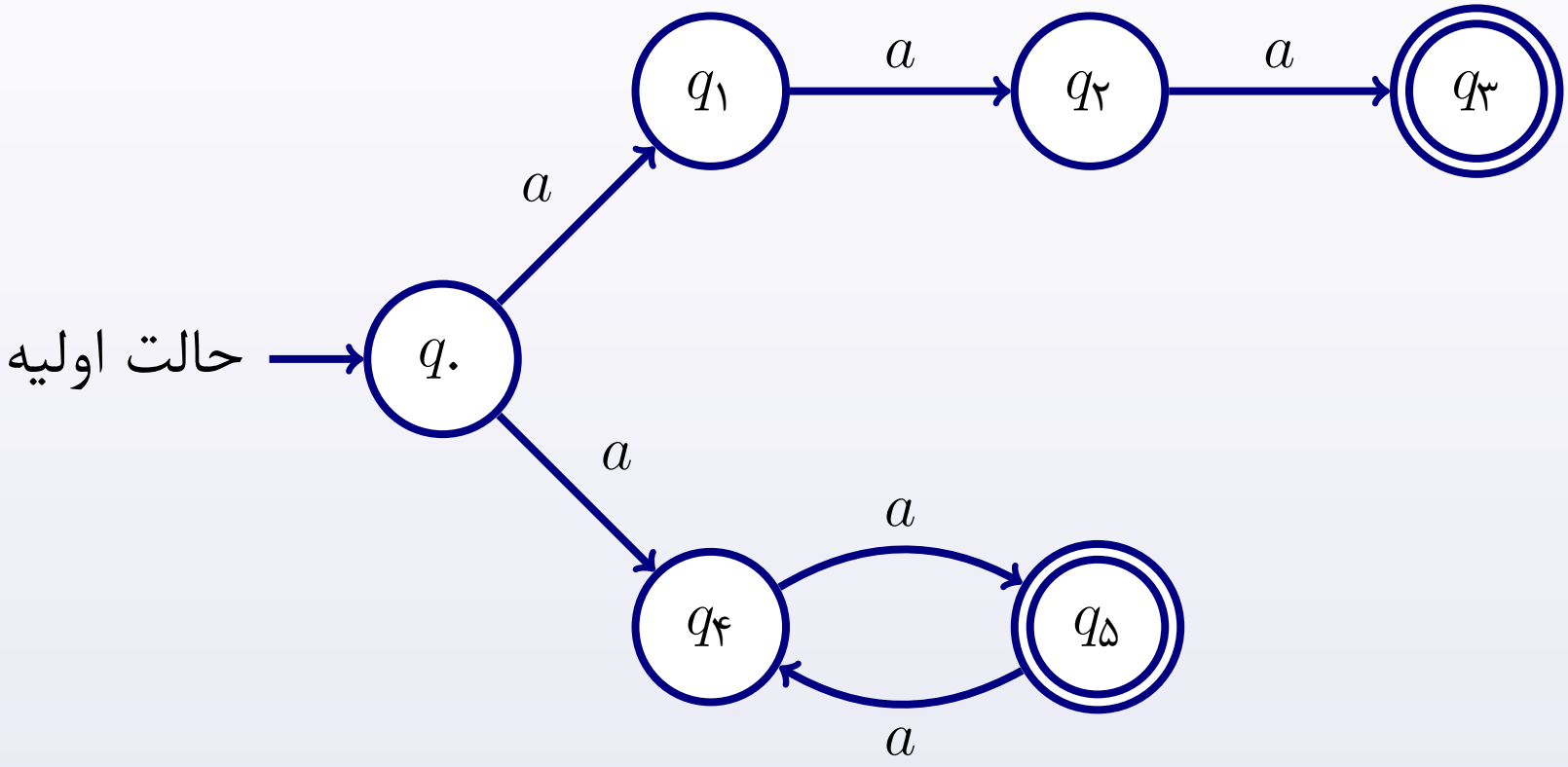
سه تفاوت بین NFA و DFA

۱. در پذیرنده نامعین برد تابع δ زیر مجموعه‌ای از مجموعه ۲^Q است؛ بنابراین مقدار آن یک عنصر از Q نیست بلکه زیر مجموعه‌ای از آن است. این زیر مجموعه، مجموعه تمام حالت‌های است که می‌توان با یک نشانه مفروض به آنها رسید.

۲. نشانه λ به عنوان ورودی قابل قبول است. یعنی، در NFA با λ، به عبارتی دیگر بدون هیچ ورودی، یک تغییر وضعیت انجام می‌شود.

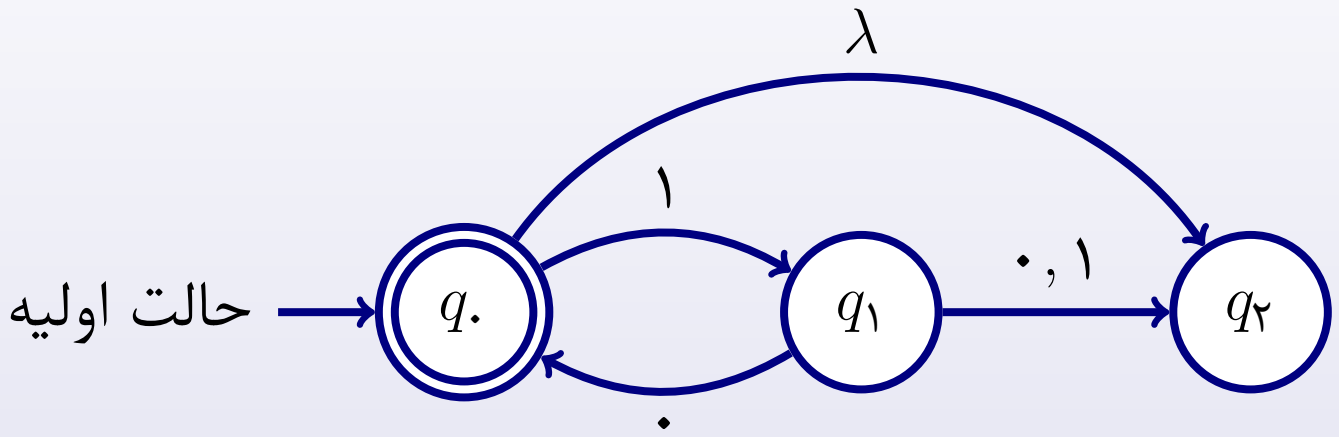
۳. نهایتاً، ممکن است که $\delta(q, a)$ تهی باشد، یعنی، هیچ تغییر وضعیتی برای نشانه a وجود ندارد..

مثال: یک NFA



از آنجا که دو یال خروجی با برچسب a از q_0 وجود دارد، این گراف گذار یک NFA را نشان می‌دهد.

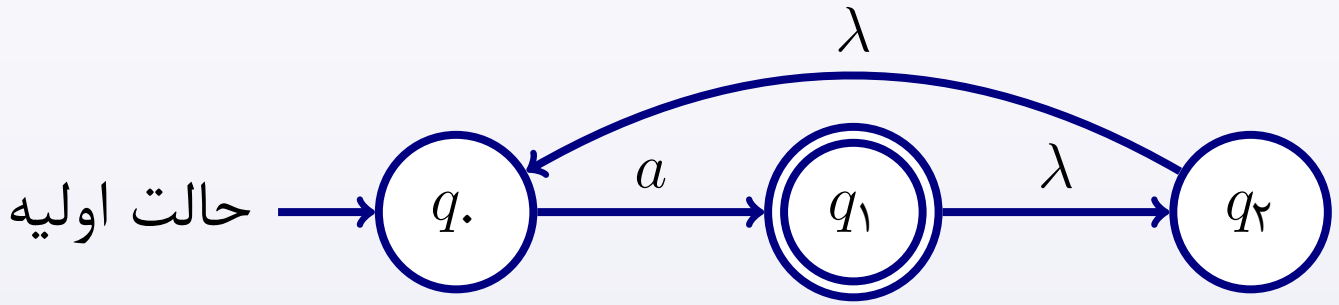
مثال: یک آتاماتای نامعین با انتقال λ



تعریف: تابع گذر توسعه یافته یک NFA: تابع $\delta^*(q_i, w)$

شامل q_j است اگر و تنها اگر یک راه در گراف گذار از q_i به q_j برای هر $q_i, q_j \in G$ و $w \in \Sigma$ با برچسب w وجود داشته باشد.

مثال: در NFA زیر چندین گذر λ و تعدادی گذر تعریف نشده، مانند $\delta(q_2, a)$ وجود دارد. حال، قصد داریم نتیجه $\delta^*(q_1, a)$ و $\delta^*(q_2, \lambda)$ را بیابیم.



یک راه با برچسب a از q_1 به خودش با وجود دو گذر λ وجود دارد: $q_1 \xrightarrow{\lambda} q_2 \xrightarrow{\lambda} q_0$. بنابراین

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

با توجه به وجود یال λ بین $q_۲$ و $q_.$، نتیجه $\delta^*(q_۲, \lambda)$ شامل $q_.$ خواهد بود. همچنین از آنجا که هر حالت بدون اینکه حرکت انجام شود در دسترس است، در نتیجه $\delta^*(q_۲, \lambda)$ شامل $q_۲$ نیز خواهد بود، لذا:

$$\delta^*(q_۲, \lambda) = \{q_., q_۲\}$$

در یک NFA، طول راه برچسب شده برای ساخت رشته w با طول آن رشته برابر نیست. برای نمونه در NFA پیشین راه برچسب شده با a طول چهار دارد.

حداکثر طول مسیر برای ساخت یک رشته

- اگر بین دو راس v_i و v_j راهی با برچسب w وجود داشته باشد باید راهی با طول کمتر از $\alpha + (1 + \alpha)|w|$ با برچسب w وجود داشته باشد. (α تعداد یال‌های λ در گراف است.)

به عبارتی دیگر، اگر مسیری از v_i به v_j وجود داشته باشد که رشته w حاصل آن باشد، آنگاه مسیری با طول $\alpha + (1 + \alpha)|w|$ یا کمتر بین این دو راس وجود دارد.

- بنابراین، برای محاسبه $\delta^*(q_i, w)$ می‌توانیم تمامی راه‌های که حداکثر $\alpha + (1 + \alpha)|w|$ طول دارند را محاسبه و از بین آنها راهی که برچسب w دارد را انتخاب کنیم.

تعریف: زبان پذیرفته شده توسط NFA ی مانند $M = (Q, \Sigma, \delta, q., F)$

مجموعه تمام رشته‌های پذیرفته شده توسط M می‌باشد. به صورت رسمی:

$$L(M) = \{w \in \Sigma^* : \delta^*(q., w) \cap F \neq \emptyset\}.$$

به عبارتی دیگر، تمامی رشته‌های مانند w که یک راه با برچسب w از راس اولیه به بعضی از راس‌های پایانی وجود داشته باشد.

- کامپیوترهای رقمی کاملاً معین هستند: وضعیت آنها در هر زمان از روی ورودی و وضعیت اولیه قابل پیش بینی است.

- عدم قطعیت روشی موثر برای توصیف بعضی زبان‌های پیچیده است.

- یک گرامر نیز می‌تواند عدم قطعیت داشته باشد؛ برای نمونه، گرامر $S \to aSb|\lambda$ می‌تواند رشته‌های متعددی را تولید کند: هر بار یکی از قانون را انتخاب می‌کنیم.

- خواهیم دید که تفاوت عمده‌ای بین DFA و NFA وجود ندارد. در نتیجه، استفاده از عدم قطعیت بحث‌های صوری را بدون تاثیر بر کلیت نتایج آسان می‌کند.

هم ارزی پذیرنده‌های قطعی معین و نامعین

در این قسمت به این پرسش پاسخ می‌دهیم:

آیا زبان‌های پذیرفته شده توسط DFA ها همان‌های هستند که توسط NFA ها پذیرفته می‌شوند؟

در آغاز به تعریف هم ارزی زبان‌ها نیاز داریم:

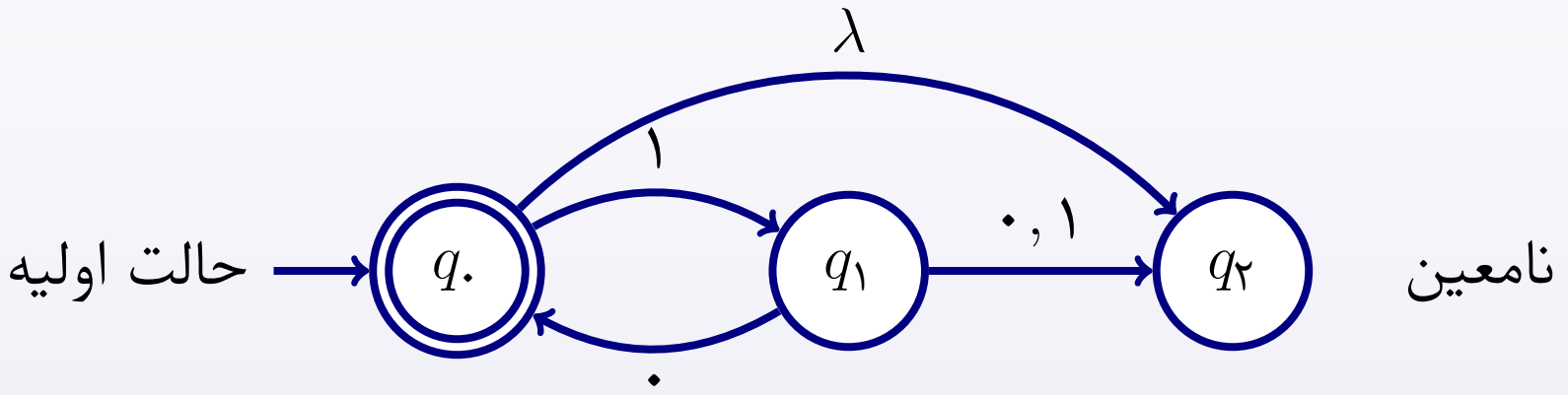
تعریف: آتاماتاهای هم ارز

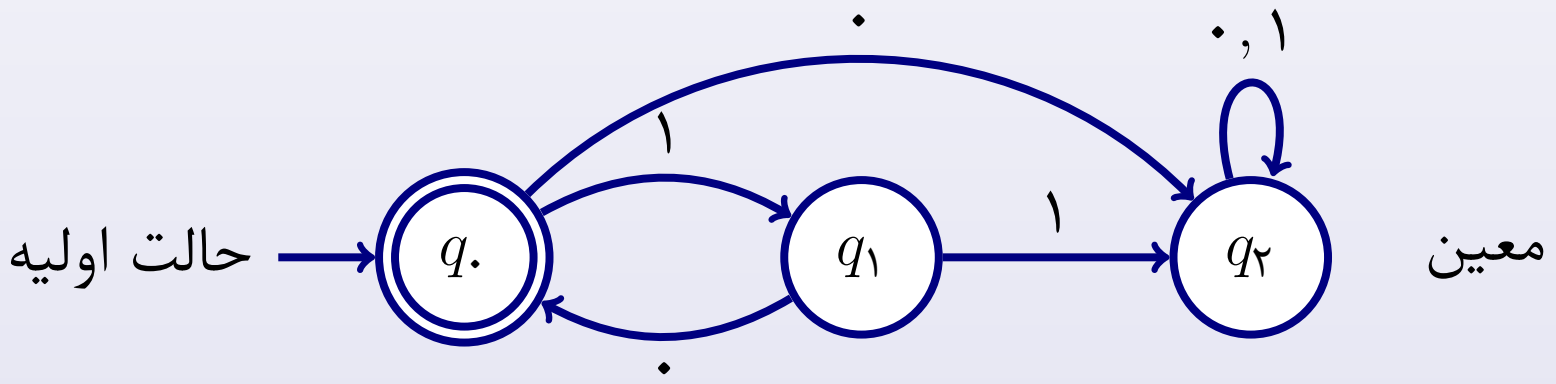
دو آتاماتای M_1 و M_2 در صورتی هم ارز هستند که

$$L(M_1) = L(M_2)$$

یعنی، دو آتاماتا هم ارزند اگر هر دو یک زبان را بپذیرند.

مثال: آتاماتای نامعین هم ارز با یک آتاماتای معین: هر دو آتاماتا زبان $\{(۱۰)^n : n \geq ۰\}$ را می‌پذیرند.





- واضح است که زبان‌های پذیرفته شده توسط DFA ها توسط NFA ها نیز پذیرفته می‌شود. زیرا، یک DFA یک NFA نیز است. غیر قطعی بودن NFA این تصور را بوجود می‌آورد که زبان‌های وجود دارد که می‌توان برای آنها NFA رسم کرد ولی نمی‌توان توسط DFA ها آن زبان‌ها را پذیرفت. **روالی که در ادامه خواهد آمد نشان می‌دهد که این تصور درست نیست.**

روال تبدیل NFA به DFA

۱. گراف G_D با راس $q.$ را ساخته و آن را راس ابتدایی در نظر بگیرید.

۲. گام‌های زیر را آنقدر تکرار کنید تا هیچ یالی فراموش نشود.

   ۱. هر یک از راس‌های $\{q_i, q_j, \cdots, q_k\}$ از G_D را که یال خروجی $a \in \Sigma$ ندارد را در نظر بگیرید.

   ۲. گذرهای $\delta^*(q_i, a)$، $\delta^*(q_j, a)$، $\cdots$، و $\delta^*(q_k, a)$ را محاسبه کنید.

   ۳. سپس، اجتماع تمام این δ^* را پیدا کنید:

$$\{q_l, q_m, \cdots, q_n\} \leftarrow \delta^*(q_i, a) \cup \delta^*(q_j, a) \cup \cdots \cup \delta^*(q_k, a)$$

۴. راسی با نام $\{q_\ell, q_m, \cdots, q_n\}$ را اگر از قبل وجود ندارد به G_D اضافه کنید.

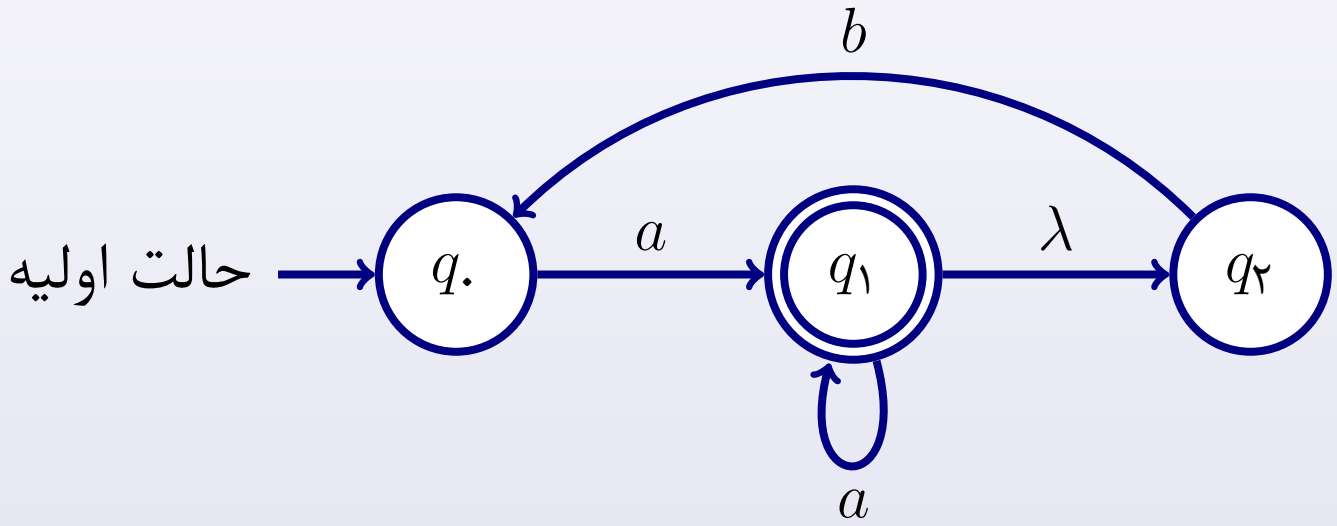
۵. یالی از $\{q_i, q_j, \cdots, q_k\}$ به $\{q_\ell, q_m, \cdots, q_n\}$ با برچسب a رسم کنید.

۳. هر یک از حالت‌های G_D که نام آنها $q_f \in F$ دارد را به عنوان یک حالت نهایی علامت گذاری کنید.

۴. اگر M_N رشته λ را می‌پذیرد، آنگاه، راس $\{q.\}$ را به عنوان راس نهایی علامت گذاری کنید.

کاربرد روال تبدیل NFA به DFA

مثال: شکل زیر یک **NFA** را نشان می‌دهد. با کمک الگوریتم گفته شده **DFA** معادل آن را بیابید.



- این NFA با q_0 شروع می‌شود، بنابراین وضعیت شروع DFA باید q_0 باشد.

- با خواندن یک a آتاماتا به حالت q_1 می‌رود؛ از آنجا نیز می‌تواند با یک حرکت λ به q_2 برود. بنابراین، DFA معادل باید حالتی با نام $\{q_1, q_2\}$ و گذر

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

داشته باشد.

- در وضعیت q_0 هیچ گذری برای ورودی b تعریف نشده است؛ یعنی:

$$\delta(\{q_0\}, b) = \emptyset.$$

حالت $\emptyset$ تعریف نشده است و معنای عدم قبول رشته دارد. بنابراین، در DFA این حالت باید یک تله غیر نهایی باشد.

- در بند پیشین حالت $\{q_۱, q_۲\}$ به DFA افزوده شد. در اینجا باید انتقال‌های آن را معین کنیم. این حالت معادل با دو حالت در NFA است، لذا، هنگام محاسبه

$$\delta(\{q_۱, q_۲\}, \alpha \in \{a, b\})$$

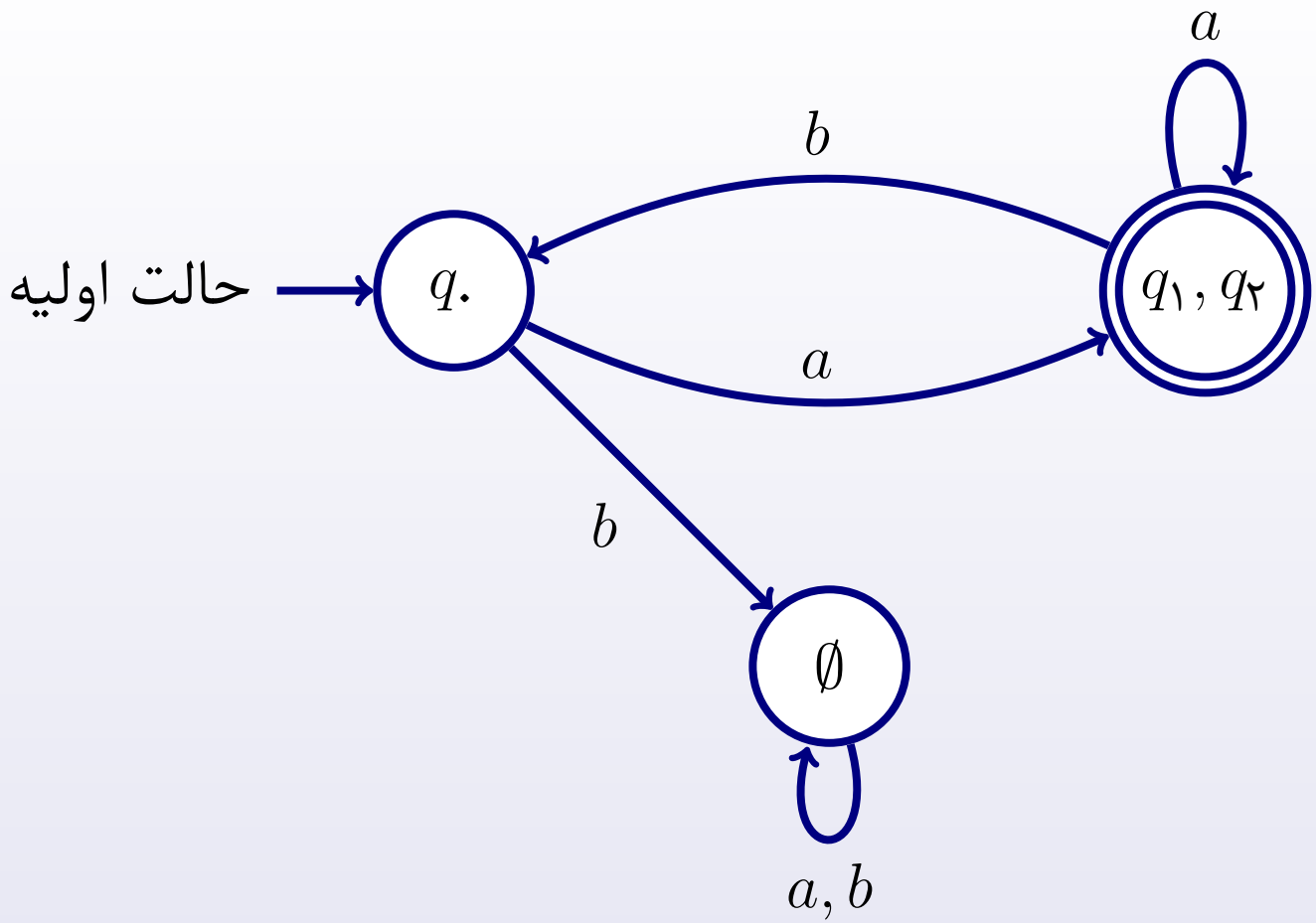
باید این دو حالت در نظر گرفته شوند: اگر NFA در حالت $q_۱$ حرف a را بخواند به $q_۱$ می‌رود؛ از $q_۱$ نیز می‌تواند با یک حرکت λ به $q_۲$ برود و از آنجا به هیچ جایی نمی‌تواند برود؛ در نتیجه:

$$\delta(\{۱۰, q_۲\}, a) = \{q_۱, q_۲\}.$$

با همین استدلال:

$$\delta(\{q_1, q_2\}, b) = \{q.\}.$$

- در NFA اولیه تمامی رشته‌های مانند w که $\delta^*(q., w)$ عضوی از F را داشته باشد پذیرفته می‌شوند. بنابراین در DFA معادل هر حالتی که عضوی از F را داشته باشد پایانی است. در این مثال، هر حالتی که q_1 در نام آن باشد پایانی است.
- تا به اینجا برای هر حالت تمام انتقال‌های آن تعریف شده است؛ کار تمام است. نتیجه در DFA زیر آمده است. این DFA معادل NFA اولیه است.


a
b
حالت اولیه
q_0
q_1, q_2
a
b
$\emptyset$
a, b

قضیه: اگر L زبان پذیرفته شده توسط $M_N = (Q_N, \Sigma, \delta_N, q., F_N)$ باشد، آنگاه یک DFA وجود دارد که $M_D = (Q_D, \Sigma, \delta_D, q., F_D)$ و

$$L = L(M_D).$$