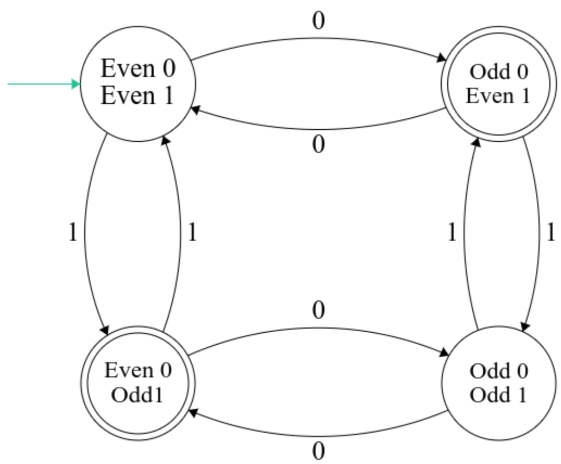
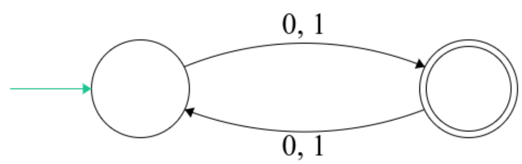
مینیمالیست

در این متن قصد داریم مسأله‌ی *کمینه‌سازی* DFA ها را بررسی کنیم. برای شروع بحث، DFA زیر را نگاه کنید:



زبان این DFA شامل رشته‌های متشکل از صفر و یک است؛ به طوری که در آن ها زوجیت (فرد یا زوج بودن) تعداد صفر ها با زوجیت تعداد یک ها متفاوت باشد. برای این منظور در هر استیت این اتوماتون، زوجیت صفرها و یک‌هایی که تا کنون خوانده شده‌اند نگه‌داری می‌شود.

حالا این یکی را نگاه کنید:



این یکی برای تشخیص رشته های به طول فرد از صفر و یک رسم شده.

همانطور که احتمالا تا الآن دریافته‌اید، یک رشته‌ی متشکل از صفر و یک دقیقاً هنگامی طولش فرد است که زوجیت تعداد یک‌ها و صفرها در آن متفاوت باشد. بنابراین این دو اتوماتون زبان یکسانی را تشخیص می‌دهند؛ در حالی که تعداد استیت های متفاوتی دارند.

برای این زبان نمیتوان DFAای با کمتر از دو استیت رسم کرد. پس کمترین تعداد استیت لازم برای رسم DFA برای آن، دو عدد است.

مسألهٔ کمینه سازی DFA به همین موضوع می‌پردازد:

> با فرض این که یک DFA مانند D در اختیار داریم، یک DFA دیگر رسم کنید که همان زبان $L(D)$ را تشخیص دهد و تعداد استیت هایش کمترین تعداد ممکن باشد. (در صورتی که چند DFA متفاوت با کمترین تعداد استیت وجود داشت، یکی از آن ها را به دلخواه انتخاب کنید.)

استیت های خارج از دسترس

۱

فرض کنید اتوماتون D برای کمینه‌سازی به ما داده شده است و در آن، استیتی مانند P به هیچ طریقی از استیت شروع قابل دسترسی نباشد. در این صورت حذف آن از اتوماتون هیچ تغییری در زبانی که میپذیرد ایجاد نخواهد کرد. پس نخستین کاری که می‌توانیم برای کاهش استیت‌ها انجام بدهیم، حذف همه‌ی استیت‌های خارج از دسترسِ استیت شروع است.

پرسش ۱: فرض کنید بعد از انجام این عملیات، یکی از استیت ها مانند P حذف شده است. استیت Q نیز در اتوماتون اولیه با خواندن حرف $a \in \Sigma$ به P میرود. حال که بعد از انجام این عملیات P حذف شده، استیت Q با خواندن a باید به کجا برود؟

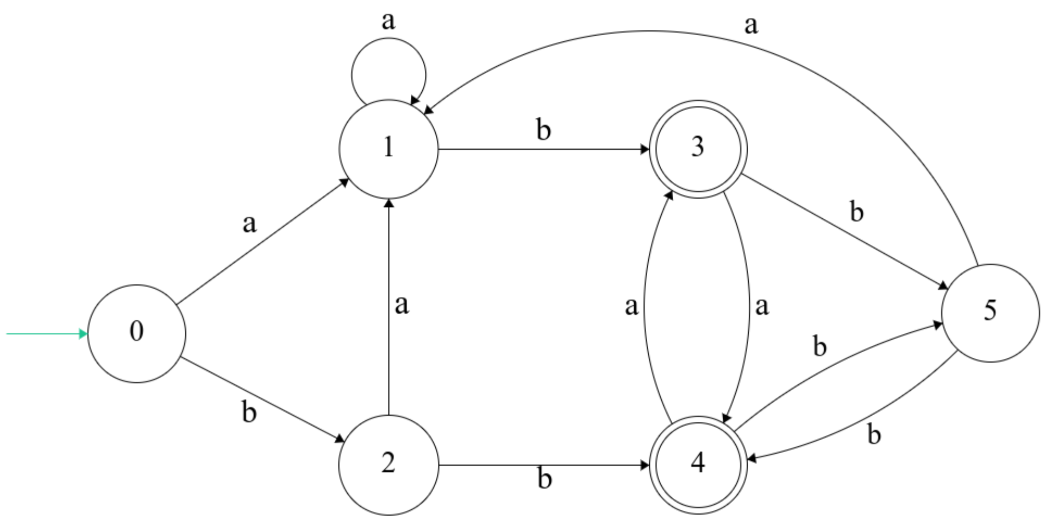
▼ راهنمایی

ثابت کنید در این شرایط، Q نیز از استیت شروع غیر قابل دسترس بوده و بنابراین آن هم در این عملیات حذف می‌شود. در نتیجه مشکل اشاره شده هرگز پیش نمی‌آید.

در ادامه ی این بحث به سادگی فرض می‌کنیم که DFA ورودی، استیت خارج از دسترس ندارد.

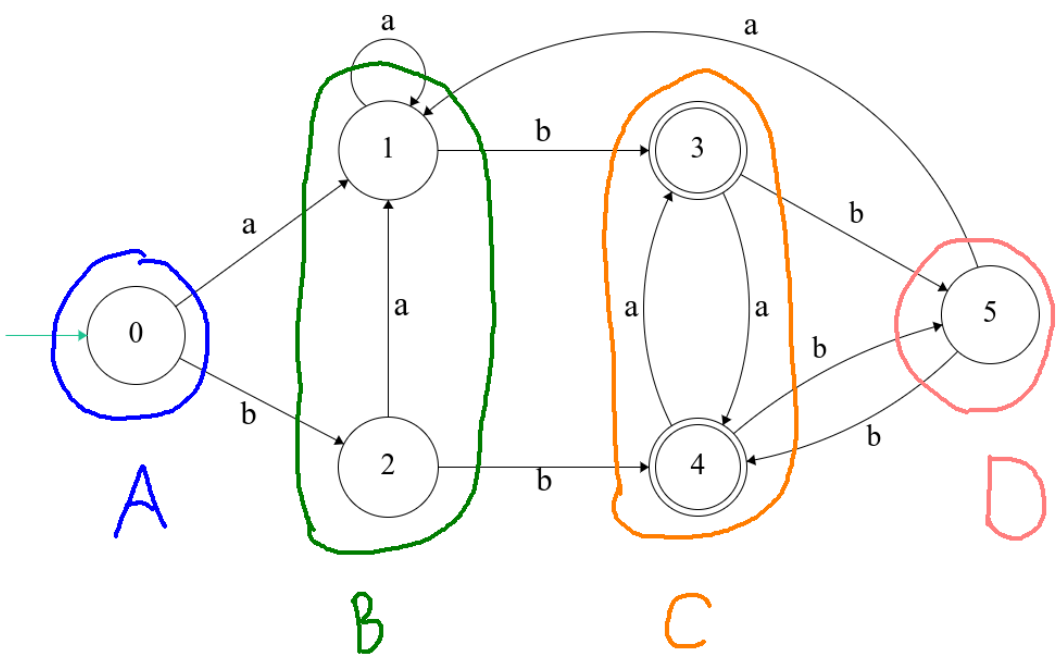
گروه‌بندی استیت‌ها

فرض کنید همه ی استیت های خارج از دسترس را از D حذف کرده باشیم و نتیجه به شکل زیر در آمده باشد:



با توجه به نحوه‌ی چینش استیت‌ها در صفحه می‌توان دید که یک نوع شباهت اساسی بین استیت‌های ۱ و ۲ وجود دارد. همینطور بین استیت های ۳ و ۴. این مسأله احتمالا راهی را برای فشرده‌تر کردن این اتوماتون در اختیار ما میگذارد.

اگر به صورت زیر استیت ها را گروه بندی کنیم



اتوماتون به دست آمده خواص جالبی خواهد داشت. می‌توانیم این اطلاعات را در مورد آن درستی‌سنجی کنیم:

- *از یک گروه خاص شروع می‌کنیم*: در شروع پردازش یک ورودی، همیشه در گروه A قرار داریم.
- *گروه فعلی، گروه بعدی را مشخص می‌کند*: اگر بدانیم که در استیت فعلی در کدام گروه هستیم، و حرف بعدی ورودی چیست، به طور یکتا مشخص می‌شود که در استیت بعدی در کدام گروه خواهیم بود. مثلاً اگر در هر کدام از استیت های گروه B باشیم و حرف بعدی ورودی b باشد، بایستی به استیتی در گروه C برویم. یا مثلاً اگر باز در همان گروه B باشیم و حرف بعدی ورودی a باشد، در همان گروه B باقی خواهیم ماند.
- *گروه فعلی، وضعیت پذیرش را مشخص می‌کند*: همه‌ی استیت‌های همگروه، از لحاظ پایانی بودن یا نبودن وضعیت یکسانی دارند. به عبارت دیگر، در یک گروه یا همه‌ی استیت‌ها استیت پایانی هستند (مثل گروه C) یا هیچ‌کدام پایانی نیستند (مثل بقیه‌ی گروه‌ها). پس صرفا با دانستن گروهی که استیت فعلی متعلق به آن است، معلوم می‌شود که اکنون در یک استیت پایانی هستیم یا نه.

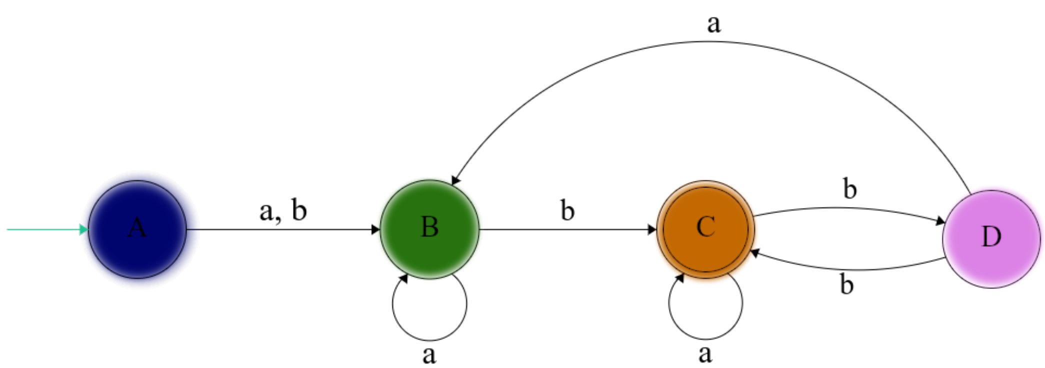
یکبار دیگر به هر سه ویژگی توجه کنیم:

۱. از یک گروه خاص شروع می‌کنیم.

۲. گروه فعلی، گروه بعدی را مشخص می‌کند.

۳. گروه فعلی، وضعیت پذیرش را مشخص می‌کند.

با وجود این سه ویژگی میتوانیم یک DFA رسم کنیم که چگونگی تغییر گروه ها را نشان میدهد:



می‌توانیم اسم آن را *اتوماتون گروهی* بگذاریم.

اگر علاقه‌مند باشیم که اتوماتون D پس از پردازش یک رشته ی خاص (مثلاً $aabbab$) به استیتی در کدام گروه می‌رسد، می‌توانیم آن رشته را در اتوماتون بالا اجرا کنیم و ببینیم استیت پایانی، متناظر با کدام گروه است. به عبارت دقیق‌تر، اگر گروه هر استیت در D مانند p را $G(p)$ بنامیم، می‌دانیم که اگر در D پردازش رشته‌ای چون w به p برسد، در اتوماتون بالا پردازش w به $G(p)$ خواهد رسید.

پرسش ۲: آیا عکس نتیجه ی فوق هم برقرار است؟ فرض کنید p استیتی در اتوماتون D باشد و $G(p)$ استیت گروه آن در اتوماتون گروهی. به فرض این که پردازش رشته‌ای چون w در اتوماتون گروهی به $G(p)$ ختم شود، آیا می‌توان حتماً نتیجه گرفت که پردازش w در D به p ختم می‌شود؟

این نتیجه به همراه ویژگی شماره ی ۳ که پیش‌تر بیان شد، نشان می‌دهد که زبان اتوماتون اصلی با اتوماتون گروهی یکسان است. پس توانستیم راه دیگری برای کوچک کردن D پیدا کنیم.

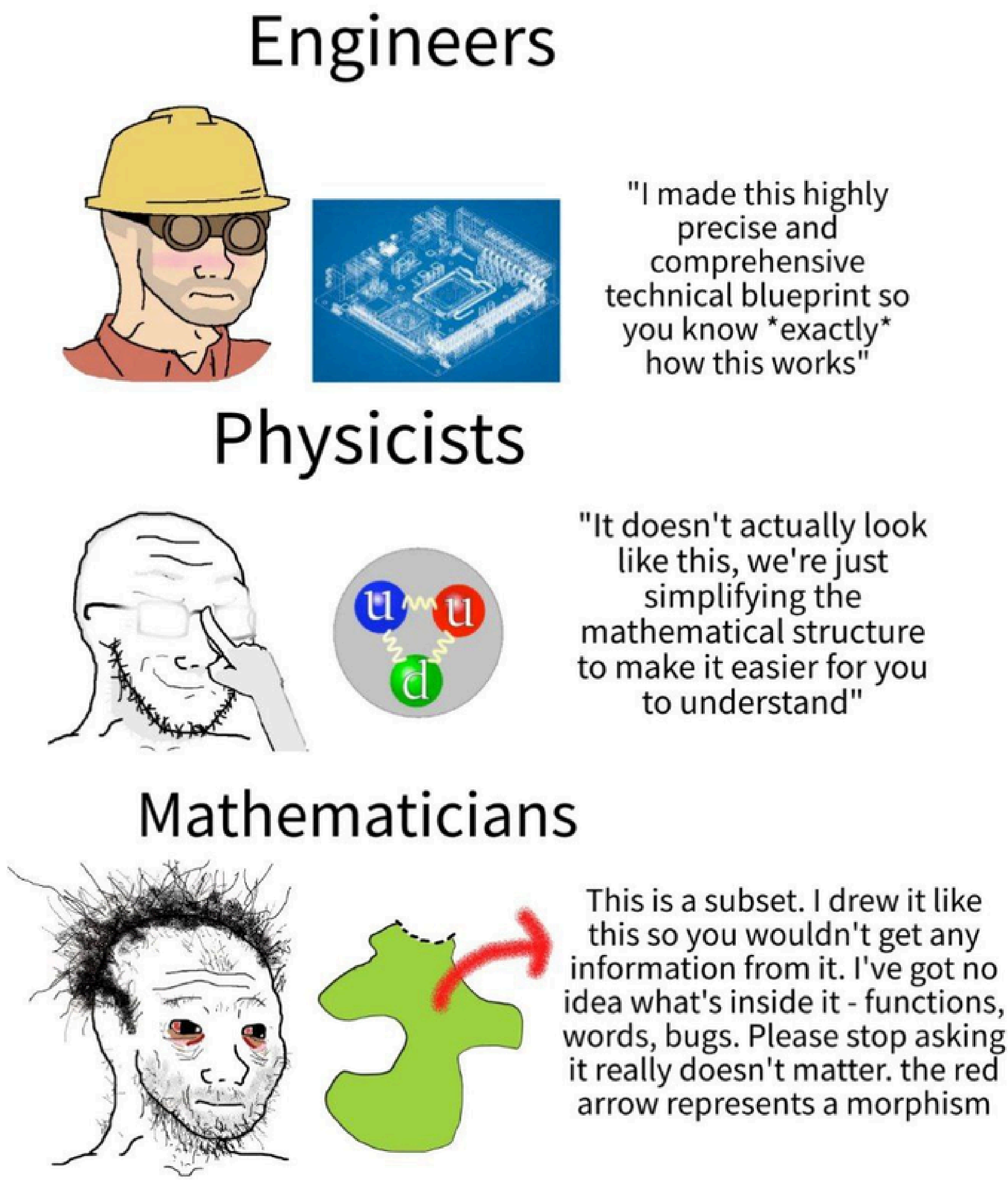
قضیه

> اگر استیت های اتوماتون D به طوری گروه‌بندی شوند که ویژگی های ۱ و ۲ و ۳ برقرار باشند، اتوماتون گروهی رسم شده برای آن گروه‌بندی، همان زبان D را تشخیص می‌دهد.

پرسش ۳: قضیه ی بالا را به صورت فرمال بیان و اثبات کنید.

پرسش ۴: در چه شرایطی تعداد استیت های اتوماتون گروهی با اتوماتون اصلی برابر است؟

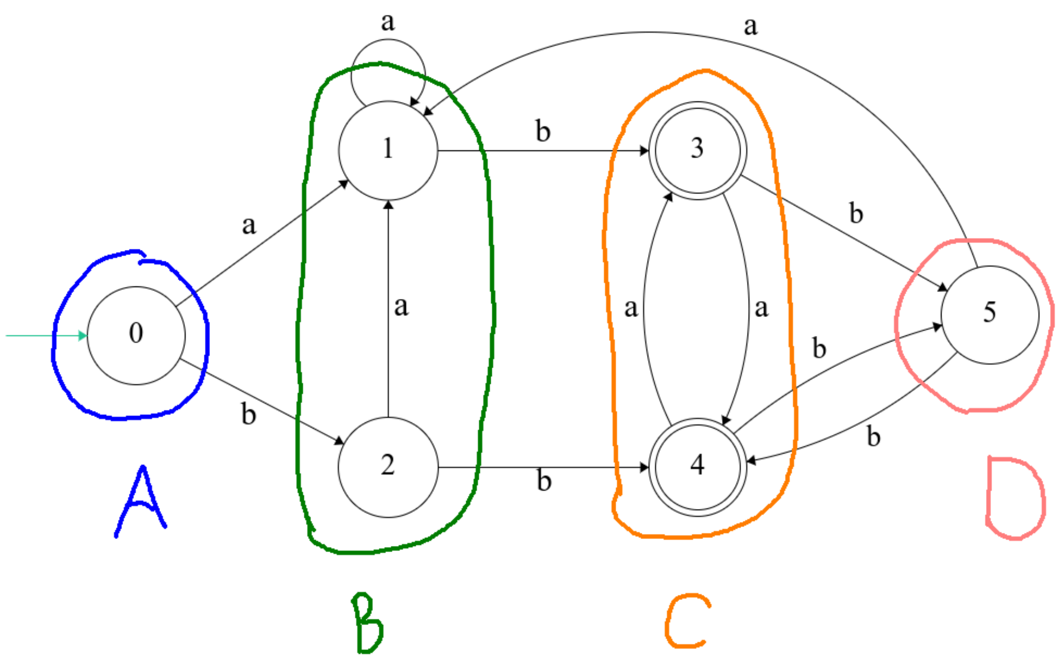
اطلاعات بیشتر! روندی که در این بخش دیدیم انتزاع (Abstraction) نام دارد و تکنیکی کاربردی در بررسی انواع اتوماتون‌ها به شمار می‌رود؛ شامل DFA ها و انواع دیگری که در این درس به آنها پرداخته نمی‌شود. به طور کلی در این رویکرد جزئیات بی‌اهمیت اتوماتون اولیه دور ریخته میشوند و اطلاعات مهم آن با یک اتوماتون دیگر (که کوچکتر و کار با آن راحت‌تر است) مدل می‌شود. به استیت های اتوماتون اصلی، استیت های عینی (Concrete) و به استیت های اتوماتون انتزاعی، استیت‌های انتزاعی (Abstract) می‌گویند. می‌توان گفت که انتزاع در سرتاسر ریاضیات رویکردی بسیار اساسی است.



رویکردی سیستماتیک

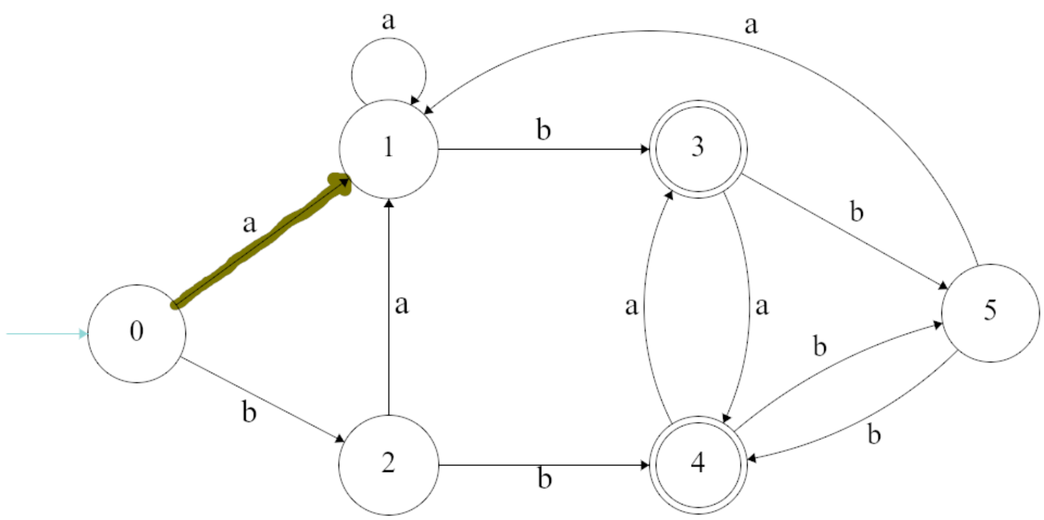
قضیه ای که بیان کردیم نشان می‌دهد که بعد از یک گروه بندی مناسب، اتوماتون گروهیِ به دست آمده با اتوماتون اصلی هم‌ارز است؛ ولی در این مورد توضیح نمی‌دهد که چگونه یک گروه‌بندی مناسب پیدا کنیم.

برای پیدا کردن یک روش سیستماتیک برای گروه‌بندی، لازم است که خواص استیت‌های همگروه را به طور دقیق‌تری بررسی کنیم. برای این کار بد نیست که دوباره به گروه‌بندی اتوماتون قبلی نگاهی بیندازیم:

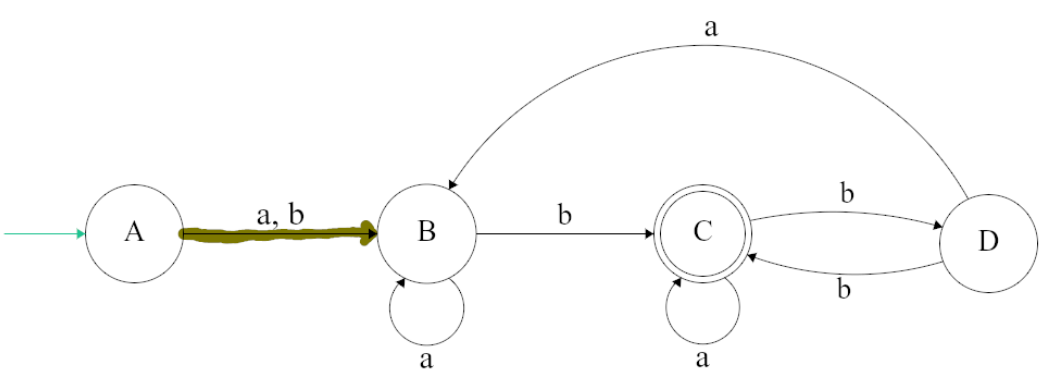


همانطور که پیش از این دیدیم، بین استیت‌های همگروه نوعی مشابهت ویژه وجود دارد. با دقت در اتوماتون گروهی شهود بهتری از این مشابهت به دست می‌آوریم.

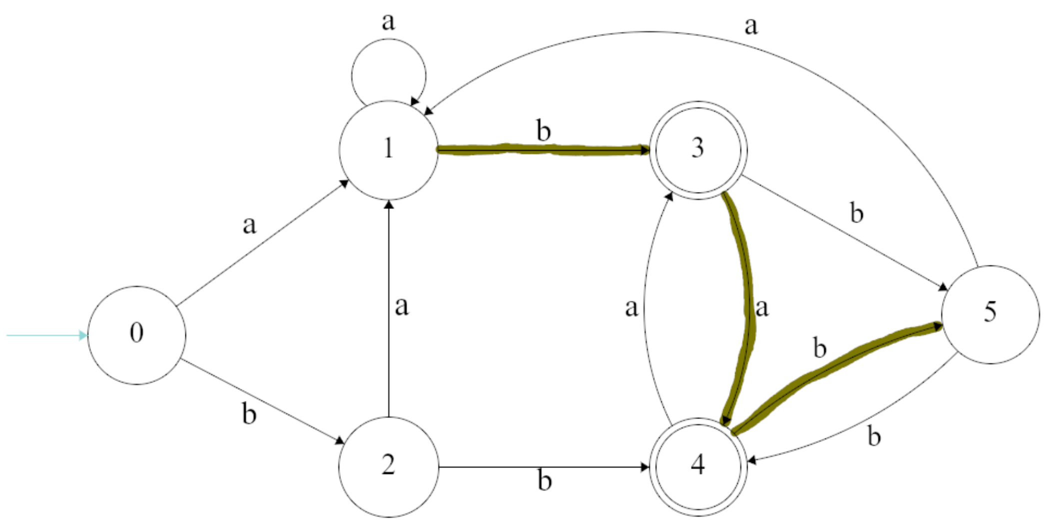
فرض کنید در اتوماتون اصلی پردازش را شروع کرده و به عنوان اولین حرف ورودی، حرف a را خوانده‌ایم. باقی حرف های ورودی هم هنوز دیده نشده اند. برای راحتی، رشته‌ی باقیمانده را s می‌نامیم. در حال حاضر در استیت ۱ قرار داریم.



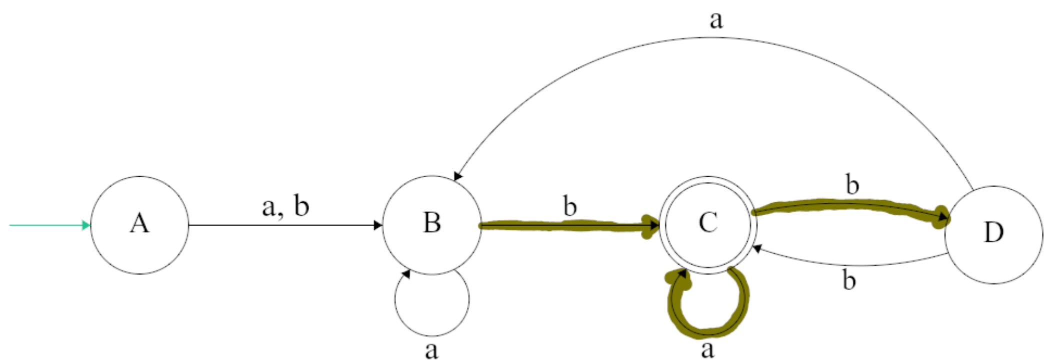
اگر پردازش را در اتوماتون گروهی شروع میکردیم اکنون در استیت B می‌بودیم.



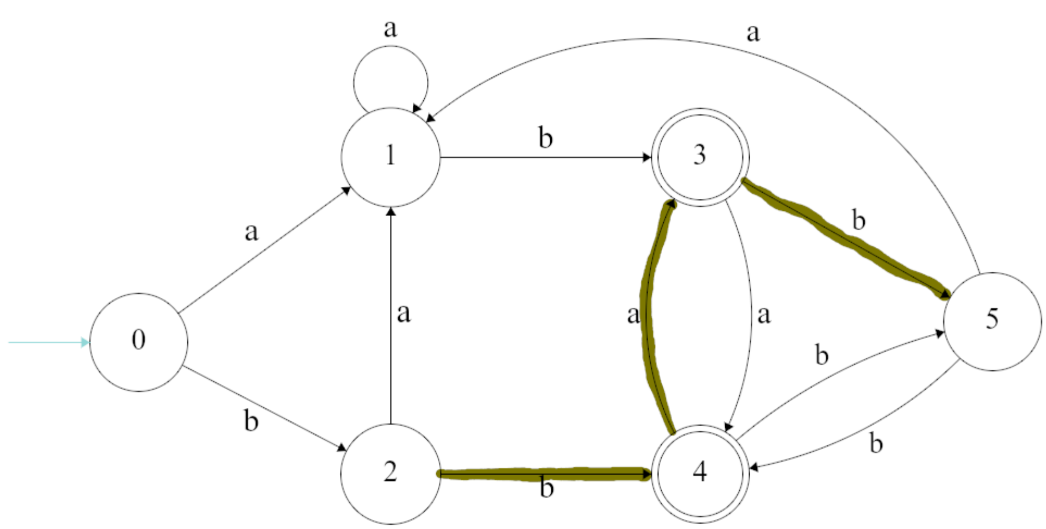
حال فرض کنید پردازش را در همان اتوماتون اصلی ادامه دهیم و بعد از دیدن رشته‌ی s به استیت p برسیم.



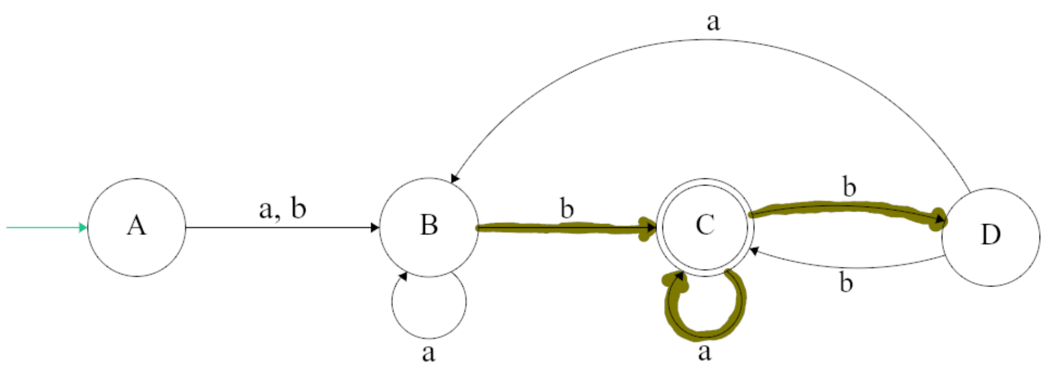
برای این که ببینیم استیت p در چه گروهی قرار دارد، می‌توانیم در اتوماتون گروهی از B شروع کنیم و s را پردازش کنیم تا ببینیم به استیت مربوط به کدام گروه می‌رسیم.



همین بررسی را در مورد استیت ۲ می‌توان انجام داد.



اگر بخواهیم بدانیم که بعد از پردازش رشته ی bs به استیتی در کدام گروه می‌رسیم، کافیست در اتوماتون گروهی از B شروع کنیم و s را پردازش کنیم تا ببینیم به استیت مربوط به کدام گروه می‌رسیم. (عین همان کاری که در پاراگراف قبلی انجام دادیم!)



در این جا به یک پدیده جالب پی‌میبریم: به ازای هر رشته مانند s، $\delta^*(1,s)$ با $\delta^*(2,s)$ همگروه است. از آن جا که استیت های همگروه از لحاظ پایانی بودن وضعیت یکسان دارند، یا هر دوی $\delta^*(1,s)$ و $\delta^*(2,s)$ پایانی اند یا هیچ یک پایانی نیستند.

استیت‌های معادل

خاصیتی که در بالا برای استیت های ۱ و ۲ کشف کردیم را می‌توان به‌طور کلی بین هر دو استیت از یک اتوماتون فرضی در نظر گرفت.

تعریف فرض کنید p و q دو استیت در یک DFA باشند. گوییم p و q با هم معادلند در صورتی که به ازای هر رشته چون s، دقیقاً هنگامی با شروع از p و پیمایش s به یک استیت پذیرش برسیم که با شروع از q

و پیمایش s هم همین اتفاق بیفتد. به بیان فرمال، در یک اتوماتون قطعی متناهی مانند

$$D = (Q, \Sigma, \delta, q_0, F)$$

گوییم که $p, q \in Q$ دو استیت معادل هستند اگر

$$\forall s \in \Sigma; \delta^*(p, s) \in F \iff \delta^*(q, s) \in F$$

از طرف دیگر، اگر p و q دو استیت معادل نباشند، رشته‌ای وجود خواهد داشت که با شروع از یک استیت و پردازش آن رشته به یک استیت پذیرش برسیم، ولی با شروع از آن یکی استیت و پردازش همان رشته به استیتی برسیم که استیت پذیرش نیست. در این شرایط می‌توانیم بگوییم که p و q بر سر رشته ی s با یکدیگر اختلاف دارند.

پرسش ۵: ثابت کنید رابطه ی معادل بودن بین دو استیت رابطه ی هم ارزی است و در نتیجه استیت‌ها را به تعدادی کلاس هم‌ارزی افراز میکند.

پرسش ۶: فرض کنید استیت های p و q با هم معادل اند. استیت p با دیدن حرف a به استیت x می‌رود و استیت q هم با دیدن a به y می‌رود. ثابت کنید x و y هم‌ارزند.

▼ راهنمایی

از برهان خلف استفاده کنید. نشان دهید اگر چنین نباشد، رشته‌ای پیدا میشود که p و q بر سر آن اختلاف داشته باشند.

پرسش ۷: ثابت کنید اگر دو استیت معادل باشند، وضعیت پایانی بودنشان با هم برابر است. به عبارت دیگر، یا هر دو پایانی اند یا هر دو پایانی نیستند.

▼ راهنمایی

در تعریف فرمال معادل بودن قرار دهید $s = \varepsilon$

می‌توان از کلاس‌های هم‌ارزی برای گروه‌بندی اتوماتون استفاده کرد .در این جا برای ساختن اتوماتون گروهی، یک استیت برای هر کلاس هم‌ارزی در نظر میگیریم.

۱. استیت شروع، کلاس هم‌ارزی شامل استیت آغازین اتوماتون اصلی است.

۲. برای تعیین این که هر استیت مانند V با دیدن حرف a به کدام استیت باید برود: یکی از استیت های اتوماتون اصلی که در آن کلاس هم ارزی قرار دارد را به دلخواه انتخاب می‌کنیم (به بیان دیگر، یکی از استیت های عینی متناظر با استیت انتزاعی مربوطه را در نظر میگیریم). فرض کنید که با دیدن حرف a، آن استیت عینی به استیت عینی q برود. استیت مربوط به آن کلاس هم‌ارزی که شامل q می‌شود را U در نظر بگیرید. در این شرایط تعیین میکنیم که استیت V با دیدن a به U برود.

۳. در صورتی که یک کلاس هم‌ارزی شامل استیت های پایانی در اتوماتون اصلی باشد، به عنوان استیت پایانی در نظر گرفته می‌شود.

پرسش ۸: با استفاده از لم اثبات شده نشان دهید که در مرحله ی ۲ فرقی نمیکند کدام استیت عینی را در آن کلاس هم‌ارزی انتخاب کنیم.

پرسش ۹: نشان دهید خاصیت‌های ۱ و ۲ و ۳ برای این گروه‌بندی برقرار است.

در ادامه اتوماتون به دست آمده را M می‌نامیم:

$$M = (G, \Sigma, \delta_M, g_0, F_M)$$

که در آن g_0 استیت متناظر با کلاس هم‌ارزی ای است که q_0 در آن قرار دارد.

پرسش ۱۰: ثابت کنید در اتوماتون M هیچ دو استیتی با هم معادل نیستند.

▼ راهنمایی

فرض کنید g_1 و g_2 دو استیت متمایز در M باشند که با هم معادل اند. می‌دانیم که g_1 و g_2 متناظر با دو کلاس هم‌ارزی از استیت‌های D هستند. از هر کدام از این کلاس‌های هم‌ارزی یک استیت در نظر بگیرید. ثابت کنید اگر g_1 و g_2 معادل باشند، استیت‌هایی که در نظر گرفته‌اید هم معادل می‌شوند که با توجه به این که در دو کلاس هم‌ارزی متفاوت قرار دارند امکان ندارد.

می‌دانیم که M همان زبان D را تشخیص می‌دهد. از طرف دیگر، جالب است که هیچ اتوماتون کم‌استیت‌تری از M وجود ندارد که همان زبان را بتواند تشخیص دهد! پس می‌توانیم ادعا کنیم که M پاسخی برای مسأله ی کمینه‌سازی است. چون این نکته در بحث ما اهمیت اساسی دارد اثبات آن را به طور

مستقیم می‌آوریم، اما خوب است که اگر حوصله دارید پیش از خواندن این اثبات خودتان قدری به آن بیندیشید.

▼ اثبات

ایده‌ی اثبات اگر یک DFA مانند $M' = (P, \Sigma, \delta_{M'}, p_0, F_{M'})$ با تعداد استیت‌های کمتر از M وجود داشته باشد که همان زبان M را می‌پذیرد، دو استیت معادل در M پیدا خواهد شد. با توجه به پرسش ۱۰ می‌دانیم که چنین چیزی غیر ممکن است. در نتیجه M' نمی‌تواند وجود داشته باشد و $M،$ کم‌استیت‌ترین اتوماتون قطعی است که زبان $L(D)$ را می‌پذیرد.

جزئیات اثبات فرض کرده بودیم که اتوماتون $D = (Q, \Sigma, \delta, q_0, F)$ هیچ استیت خارج از دسترسی ندارد. بنابراین M هم استیت خارج از دسترس نخواهد داشت.

▼ توضیح

یک استیت M مانند g را در نظر بگیرید. می‌دانیم که g متناظر با یک گروه از استیت‌های D است. یکی از اعضای گروه متناظر با g مانند q را در نظر می‌گیریم. از آن‌جا که q خارج از دسترس نیست، رشته‌ای مانند s وجود دارد که با شروع از استیت اولیه ی D و پیمایش s در نهایت به q برسیم. پس با شروع از استیت اولیه ی M و پیمایش همان s، به استیت متناظر با گروهِ q (یعنی g) می‌رسیم که نتیجه می‌دهد g خارج از دسترس نیست.

می‌توانیم ادعای «M استیت خارج از دسترس ندارد» را به گونه‌ی دیگری بیان کنیم: به ازای هر استیت M مانند g، رشته‌ای چون $r(g)$ پیدا می‌شود که با شروع از استیت آغازین M و پردازش $r(g)$ به استیت g برسیم. حالا از $r(g)$ استفاده می‌کنیم تا هر استیت M را به یک استیت در M' مپ کنیم.

تابع

$$\begin{cases} f : G \to P \\ f(g) = \delta^*_{M'}(p_0, r(g)) \end{cases}$$

را در نظر می‌گیریم. این تابع به ازای هر استیت M مانند g، به ما می‌گوید که اگر از استیت آغازین M' شروع کرده و $r(g)$ را پیمایش کنیم به کدام استیت خواهیم رسید. از آن‌جا که تعداد استیت‌های M' از

M کم‌تر است، طبق اصل لانه‌کبوتری دو استیت متفاوت در M پیدا می‌شوند که به استیت یکسانی در M' مپ شده باشند:

$$|P| < |G| \Rightarrow \exists g_1, g_2 \in G; f(g_1) = f(g_2), g_1 \neq g_2$$

این امر باعث می‌شود تا g_1 و g_2 معادل باشند:

$$\delta_M^*(g_1, s) \in F_M \iff \delta_M^*(\delta_M^*(g_0, r(g_1)), s) \in F_M \iff \delta_M^*(g_0, r(g_1)s) \in F_M \iff$$

چون در M هیچ دو استیتی با هم معادل نیستند، فرض خلف باطل بوده و M' نمی‌تواند وجود داشته باشد.

پس دقیقاً کجای این تمرین عملی بود؟!

خوبیِ مفهوم معادل بودن دو استیت این است که برای تشخیص آن، الگوریتم مناسبی وجود دارد.

کاری که این الگوریتم انجام می‌دهد پرداختن به مکمل این مسأله است. در واقع این الگوریتم استیت‌هایی که معادل *نیستند* را مشخص می‌کند که بر اثر آن می‌فهمیم کدام استیت ها با هم معادل *هستند*.

شیوه ی کار آن به این صورت است:

- برای هر دو استیت مانند p و q، وضعیت معادل بودنشان را نگه‌داری می‌کنیم. برای این منظور می‌توانیم مثلاً از یک آرایه ی دوبعدی استفاده کنیم. 0 بودن یک درایه در این ماتریس به این معناست که هنوز نمیدانیم p با q معادل است یا نه. 1 بودن به این معناست که فهمیده‌ایم که p و q با هم مادل *نیستند*.
- در ابتدا هیچ اطلاعاتی نداریم، بنابراین همه ی درایه های آرایه ی ما صفر است.
- در گام صفرم، استیت‌ها را دو به دو چک می‌کنیم. اگر یکی از آن‌ها استیت پذیرش بود و دیگری نبود، متوجه می‌شویم که معادل نیستند (به پرسش ۶ توجه کنید)، و درایه ی مربوط به آن دو را ۱ می‌کنیم.

- در گام اول، استیت‌ها را دو به دو چک می‌کنیم. اگر فهمیده بودیم که معادل نیستند که هیچ. در غیر این صورت، همه ی حرف های زبان (یعنی اعضای Σ) را به ترتیب در نظر می‌گیریم. اگر p و q با آن حرف به استیت های p' و q' بروند و فهمیده باشیم که p' و q' معادل نیستند، متوجه می‌شویم که p و q هم معادل نیستند (به پرسش ۶ توجه کنید) و درایه ی مربوط به آن دو را 1 می‌کنیم. آن قدر گام قبلی را تکرار می‌کنیم تا به جایی برسیم که تکرار آن دیگر هیچ درایه ای را ۱ نمی‌کند.

پرسش ۱۱: ثابت کنید الگوریتم بالا خاتمه‌پذیر است.

پرسش ۱۲: ثابت کنید اگر این الگوریتم درایه ی دو استیت را ۱ کرده باشد، آن دو استیت واقعا غیر معادل هستند.

▼ راهنمایی

نشان دهید یک رشته ی مورد اختلاف بین آن دو وجود دارد.

پرسش ۱۳: نشان دهید عکس گزاره ی بالا هم برقرار است؛ یعنی اگر دو استیت غیر معادل باشند در جایی از الگوریتم درایه ی مربوط به آن ها ۱ می‌شود.

▼ راهنمایی

دو استیت غیر معادل را در نظر بگیرید و به کوتاه‌ترین رشته ی مورد اختلاف بینشان مانند s توجه کنید. (ممکن است چند تا از این «کوتاه ترین» رشته‌های مورد اختلاف وجود داشته باشد، یکی را به دلخواه انتخاب کنید). ثابت کنید الگوریتم حداقل تا گام $|s|$ام ادامه خواهد داشت و در آن گام، درایه ی مربوط به آن دو استیت ۱ خواهد شد.

پرسش۱۴: می‌توان الگوریتم را به نحوی اجرا کرد که کوتاه‌ترین رشته ی مورد اختلاف بین هر دو استیت غیر معادل را هم بتواند خروجی دهد. چگونگی این کار را بررسی کنید.

پرسش ۱۵: می‌توان الگوریتم را به نحوی اجرا کرد که اگر چند «کوتاه‌ترین رشته ی مورد اختلاف» بین دو استیت وجود داشت، آنی را خروجی دهد که در ترتیب لغت‌نامه‌ای (Lexicographical) پیش از بقیه می‌آید. چگونگی این کار را بررسی کنید.

جاده طی شد بی‌سبب؛ به کجا باید رفت؟

در این تمرین برای هر کدام از دانشجویان یک DFA غیر کمینه در نظر گرفته شده. شما بایستی با اجرای الگوریتم اشاره شده آن را کمینه کنید و به سؤالاتی درباره‌ی آن پاسخ دهید. فراموش نکنید که پیش از اجرای الگوریتم استیت‌های خارج از دسترس را حذف کنید!

در DFA ورودی مجموعه ی الفبا به صورت

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

خواهد بود.

برای گزارش دادن DFA کمینه شده آن را به یک شیوه ی قراردادی بایستی خروجی دهید. فرض کنید می‌خواهید $M = (Q, \Sigma, \delta, q_0, F)$ را خروجی دهید. برای این کار:

۱. ابتدا تمام رشته‌های Σ^* را ابتدا بر حسب طولشان و سپس بر حسب ترتیب لغت‌نامه‌ای مرتب کرده و در دنباله ی a می‌نویسیم:

$$a = \langle \varepsilon, 0, 1, \cdots, c, 00, 01, \cdots, cc, 000, \cdots \rangle$$

۱. حال دنباله ی b را در نظر می‌گیریم که عضو iام آن، استیتی است که پس از پیمایش a_i به آن می‌رسیم.
۲. استیت‌ها را به ترتیب اولین حضورشان در دنباله ی b مرتب کرده و از ۰ شماره‌گذاری می‌کنیم. (باری دیگر توجه کنید که شماره‌گذاری را از ۰ شروع کردیم)
۳. به ترتیب شماره، استیت‌ها را در نظر می‌گیریم. اگر استیت iام برابر q باشد، دو خط زیر را به خروجی اضافه می‌کنیم:

$i :$

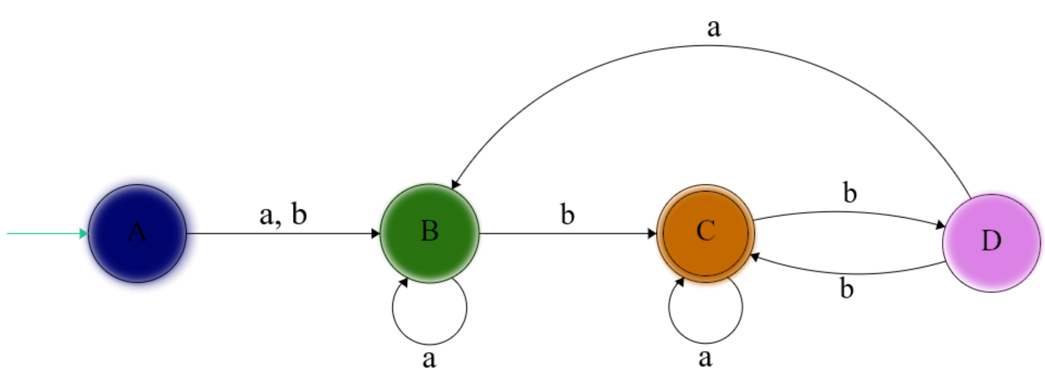
$\delta(q, 0)\delta(q, 1)\cdots\delta(q, c)$

۴. در آخرین خط، استیت‌های پایانی را به ترتیب شماره (از کوچک به بزرگ) و با یک فاصله بینشان می‌نویسیم.

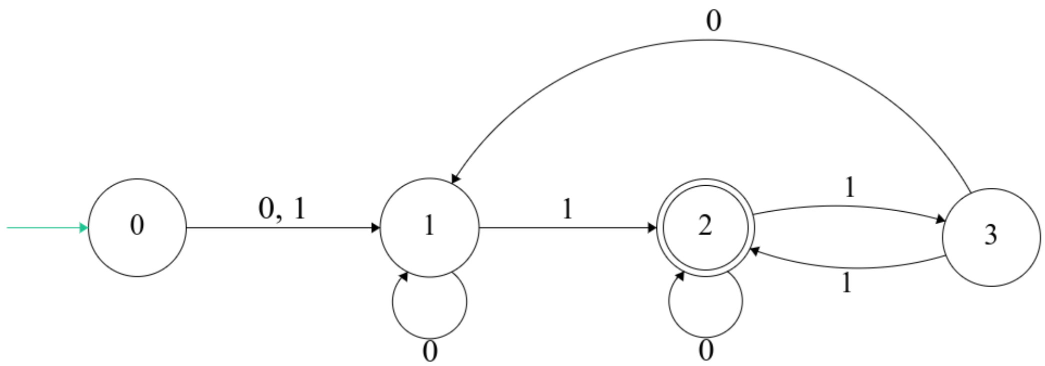
پس از خروجی دادن DFA کمینه شده، بایستی به سؤالاتی در مورد آن پاسخ دهید. همانطور که در پرسش ۱۰ دیدیم، هیچ یک از استیت‌های DFA کمینه شده با دیگری معادل نیست؛ بنابراین یک رشته‌ی مورد اختلاف بین هر دو استیت متمایز آن وجود خواهد داشت. در پرسش ۱۵ بررسی کردیم که چگونه می‌توان از الگوریتم کمینه‌سازی استفاده کرد تا کوچک‌ترین رشته‌ی مورد اختلاف بین هر دو استیت به دست بیاید. در ورودی، چند سؤال در این کوچکترین رشته ی مورد اختلاف بین استیت iام و jام از شما پرسیده شده است. سؤال ها با A و B نام‌گذاری شده اند. بعد از خروجی دادن DFA کمینه شده، بایستی پاسخ سؤال‌ها را به ترتیب خروجی دهید.

یک نمونه از ورودی و خروجی

در اوایل متن یک DFA را به صورت دستی گروه‌بندی کردیم و اتوماتون گروهی حاصل را رسم کردیم.



در این بخش برای نمونه اتوماتونی شبیه به این اتوماتون را کمینه می‌کنیم و در ضمن با فرمت ورودی و خروجی بیشتر آشنا می‌شویم. در اتوماتون این بخش به جای a و b با ۰ و ۱ کار می‌کنیم. استیت‌ها را هم از ۰ تا سه شماره‌گذاری می‌کنیم



فرض کنید اتوماتون بالا به این شکل ورودی داده شده باشد:

```
This DFA contains 4 states.
The transition function is described below:

0 --0--> 1
0 --1--> 1

1 --0--> 1
1 --1--> 2

2 --0--> 2
2 --1--> 3

3 --0--> 1
3 --1--> 2

Final state(s) of this DFA:
2
```

```
After minimizing this DFA, what would be the 'smallest'string with different r
```

```
A. 1 and 2?
B. 0 and 1?
```

برای به دست آوردن خروجی بایستی ابتدا استیت‌های خارج از دسترس را از اتوماتون حذف کنیم. در این جا تمام استیت‌ها در دسترس هستند، بنابراین به سادگی به مرحله‌ی بعدی می‌رویم.

در گام بعد بایستی استیت‌های معادل را بیابیم. برای این کار لیستی از استیت‌هایی که معادل *نبودن* آن‌ها را کشف کرده‌ایم تهیه می‌کنیم که در شروع خالی است.

$$\langle\rangle$$

می‌دانیم که استیت‌های پایانی و غیر پایانی معادل نیستند، پس می‌توانیم جفت‌های زیر را به لیست اضافه کنیم:

$$\langle(0,2),(1,2),(2,3)\rangle$$

استیت ۰ با دیدن ۱ به استیت ۱ می‌رود. استیت ۱ هم با دیدن ۱ به ۲ می‌رود. می‌دانیم که استیت ۱ با استیت ۲ معادل نیست؛ بنابراین استیت‌های ۰ و ۱ هم نمی‌توانند معادل باشند:

$$\langle(0,2),(1,2),(2,3),(0,1)\rangle$$

حال به استیت‌های ۰ و ۳ توجه می‌کنیم. $\delta(0,1)=1$ و $\delta(3,1)=2$. باری دیگر، می‌دانیم که استیت ۱ با استیت ۲ معادل نیست. پس ۰ و ۳ هم نمی‌توانند معادل باشند:

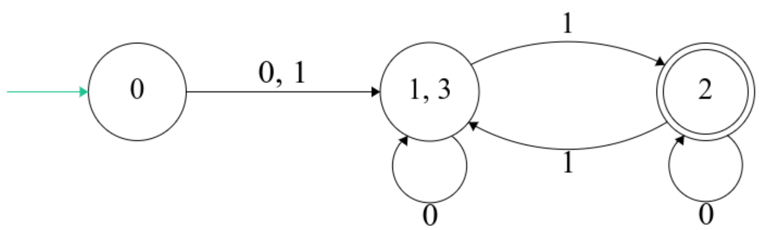
$$\langle(0,2),(1,2),(2,3),(0,1),(0,3)\rangle$$

تنها جفتی که در هنوز در لیست وارد نشده جفت (1, 3) است. با طی کردن روند بررسی روی این جفت، می‌بینیم که به لیست اضافه نمی‌شوند.

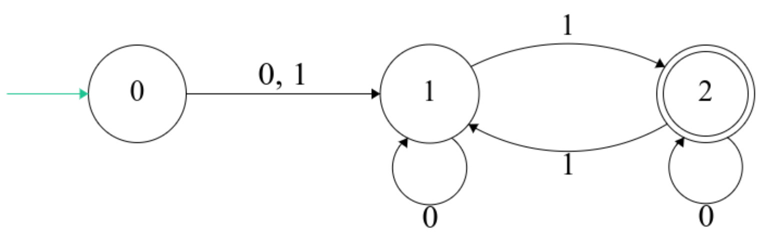
در این مرحله، هر کدام از جفت استیت‌ها یا به لیست اضافه شده اند و یا انجام بررسی روی آن ها منجر به تغییر لیست نمی‌شود. پس الگوریتم خاتمه می‌یابد و جفت های غیر معادل مشخص می‌شوند:

$$\langle (0,2), (1,2), (2,3), (0,1), (0,3) \rangle$$

استیت های ۱ و ۳ با هم معادل هستند و می‌توان آن دو را ادغام کرد. بعد از این کار اتوماتون زیر به دست می‌آید



استیت‌های آن را طبق قرارداد شماره‌گذاری می‌کنیم:



در ورودی در مورد کوچکترین رشته‌ی مورد اختلاف بین استیت ۱ و ۲ و همچنین بین استیت ۰ و ۱ پرسیده شده است. با توجه به پرسش ۱۵ می‌توانستیم در حین اجرای الگوریتم این رشته‌ها را به دست بیاوریم. در این جا چون اتوماتون کوچک است مستقیماً می‌توانیم جواب را با مشاهده بیابیم. کوچکترین رشته ی مورد اختلاف بین ۱ و ۲، رشته ی ε است. کوچکترین رشته ی مورد اختلاف بین ۰ و ۱ هم رشته ی 1 می‌باشد. بنابراین خروجی مناسب برای این مسأله به صورت زیر خواهد بود:

```
0:
1 1
1:
1 2
2:
2 1
2
A:
B:1
```

شیوه ی داوری و نمره‌دهی

در این سؤال برای هر دانشجو یک تست‌کیس متفاوت در نظر گرفته شده است. شما بایستی ابتدا به فایل تست‌کیس‌ها مراجعه کنید و با جست‌وجوی شماره ی دانشجویی خود، ورودی در نظر گرفته شده برای خود را مشاهده کنید. در گام بعد پاسخ را به دست بیاورید و در قالب یک فایل متنی در کوئرا بارگذاری نمایید.

داوری و نمره‌دهی این سؤال به صورت صفر و یکی است. برای چک کردن درستی پاسخ شما، از آن دَرهَمَک (Hash) ۲۵۶ بیتی گرفته می‌شود و با درهمک پاسخ درست مقایسه می‌شود. در صورت برابری، نمره ی کامل و در غیر این‌صورت نمره ی صفر دریافت خواهید کرد.

در کنار ورودی، درهمک پاسخ صحیح در اختیار شما قرار خواهد گرفت تا پیش از ارسال بتوانید پاسخ خود را درستی‌سنجی کنید.

نکات پایانی

- برای به دست آوردن درهمک ۲۵۶ بیتی پاسخ خود می‌توانید از دستور `sha256sum path_to_my_answer` در سیستم‌عامل لینوکس یا دستور `certutil -hashfile [file location] SHA256` در سیستم‌عامل ویندوز استفاده کنید. همچنین کتابخانه ی `hashlib` در پایتون برای این کار قابل استفاده است.
- در هنگام نوشتن پاسخ به فرمت دقیق خروجی توجه کنید. به طور خاص در مورد تعداد space و newline در هر جا دقت کافی را داشته باشید.
- اگر مایل بودید که پاسخ شما به پرسش‌های این متن به طور موردی بررسی شود، می‌توانید با آیدی MrSinaImani@ در ارتباط باشید.