به نام خدا

نظریه زبان‌ها و ماشین‌ها- بهار ۱۴۰۱

پاسخ تمرین شماره 10

دستیار آموزشی این مجموعه: صبا شهسواری

sabashahsavari@ut.ac.ir

تاریخ تحویل :

---

1) گزاره‌های زیر را اثبات یا نفی کنید.

الف) اگر زبان یک ماشین تورینگ تصمیم پذیر باشد، آن ماشین حتما توقف‌پذیر است.

ب) اگر زبان A تصمیم‌پذیر باشد و $B \subseteq A$ باشد، آن گاه B نیز تصمیم‌پذیر است.

پاسخ

الف) این جمله غلط است. برای مثال ماشین تورینگ M را در نظر بگیرید که روی هر ورودی بی‌نهایت حلقه می‌زند. زبان این ماشین تورینگ $L = \emptyset$ است. بدیهی است که این زبان تصمیم پذیر است(یک ماشین تورینگ که هر ورودی را رد میکند می‌تواند L را decide کند.) اما ماشین M هیچ وقت متوقف نمی‌شود.

ب) این جمله غلط است. می‌توان $A = \Sigma^*$ در نظر گرفت که یک زبان تصمیم‌پذیر است. اما هر زبان دیگری زیرمجموعه این زبان است و می‌دانیم برخی از زبان‌ها تصمیم‌ناپذیر هستند. برای مثال می‌توان زبان B را مسیله برابر بودن زبان دو CFG در نظر گرفت که قبلا ثابت کردیم تصمیم‌ناپذیر است.

2) زبان L شامل تمام گرامرهای خطی راستی می‌شود که رشته‌هایی را که در آن‌ها زوجیت تعداد a و b برابر است، می‌پذیرند. ثابت کنید L تصمیم‌پذیر است. $(\Sigma = \{a, b\})$

پاسخ

ابتدا گرامر خطی راست را به DFA تبدیل می‌کنیم.برای ساخت DFA، ابتدا به‌ازای هر variable یک state تعریف می کنیم. سپس برای هر production rule یک یال می‌سازیم به طوری‌که کاراکترهای آن را روی یال و variable بعدی را state مقصد در نظر می‌گیریم. در صورتی که بیش از یک کاراکتر در production rule بود، یک سری state میانی تعریف می‌کنیم. و اگر هیچ variable ای در سمت راست production rule نداشته باشیم، به state پایانی می‌رویم. این DFA را a می‌نامیم.

حال باید برای چک کردن برابر بودن زوجیت تعداد 0 و 1 ها یک DFA بسازیم. فرض کنید اسم این DFA را b بگذاریم.

می‌دانیم برای محاسبه اشتراک دو DFA یک الگوریتم پایان پذیر وجود دارد. پس حاصل $a \cap \bar{b}$ را حساب کرده و c می‌نامیم. اگر زبان c تهی باشد زبان L پذیرفته می‌شود و در غیراین صورت رد می‌شود. از آن جا که تمام عملیات‌های محاسبه اشتراک، مکمل و بررسی تهی بودن زبان یک DFA، پایان پذیر هستند پس الگوریتم فوق برای تصمیم گیری درباره L حتما خاتمه پیدا می‌کند.

3) تصمیم‌پذیری زبان زیر را بررسی کنید.

L = {<A,B,C> |A,B,C are DFAs over the same alphabet Σ, and L(A)=L(B) ∩ L(C)}

پاسخ

می‌دانیم L(A)=L(B) ∩ L(C) به این معناست که L(A) ⊆ (L(B) ∩ L(C)) و (L(B)∩L(C)) ⊆ L(A)

این دو حالت را جداگانه بررسی می‌کنیم. (همچنین می‌دانیم L(A) ⊆ (L(B) ∩ L(C)) معادل است با اینکه اشتراک مکمل((L(B) ∩ L(C) و L(A) تهی باشد)

برای تصمیم‌گیری این زبان از الگوریتم زیر استفاده می‌کنیم:

1. ابتدا چک می‌کنیم که A,B,C هر سه DFA باشند و اگر نباشند L را reject می‌کنیم.
2. می‌دانیم می‌توان با الگوریتم پایان پذیر DFA حاصل از اشتراک دو DFA را تشکیل داد. این DFA را D می‌نامیم. L(D)=L(B) ∩ L(C)
3. می‌دانیم الگوریتم‌های محاسبه مکمل و تهی بودن DFA پایان پذیر هستند. مکمل D را می‌سازیم و آن را E می‌نامیم.
4. DFA اشتراک A و E را می‌سازیم. L(F)=L(A) ∩ L(E)
5. اگر زبان F تهی نباشد به این معناست که L(A) ⊈ (L(B) ∩ L(C)) پس L را reject می‌کنیم.
6. برای بررسی حالت دوم ابتدا A را مکمل می‌کنیم و سپس DFA اشتراک آن را با D می‌سازیم و آن را G می‌نامیم.
7. اگر زبان G تهی نباشد L را رد می‌کنیم و در غیر این صورت accept می‌کنیم.

4) فرض کنید L یک زبان تصمیم‌پذیر است. نشان دهید 'L تصمیم پذیر است.

$$L' = \{w \in \Sigma^* \mid xw \in L \wedge wy \notin L\}$$

پاسخ

L یک زبان تصمیم‌پذیر است پس یک ماشین تورینگ مانند M وجود دارد که آن را decide می‌کند. برای اثبات تصمیم‌پذیر بودن 'L ، ماشین تورینگ ' M را به شکل زیر طراحی می‌کنیم:

ماشین 'M دو نوار دارد. در ابتدا ورودی w را از روی نوار ۱ به روی نوار ۲ کپی می‌کند. سپس رشته x را در نوار ۱ و قبل از رشته w می‌نویسد و رشته y را در نوار ۲ و بعد از w می‌نویسد و در نهایت head نوار ۱ را در ابتدای رشته xw و head نوار ۲ را در ابتدای رشته wy قرار می‌دهد. حال ماشین M را روی این دو نوار اجرا می‌کنیم. ماشین 'M رشته w را می‌پذیرد اگر ماشین M رشته xw را بپذیرد و رشته yw را رد کند و در غیر این صورت آن را رد می‌کند.

5) زبان L شامل زوج‌های $< G, B >$ است که در آن G یک گرامر مستقل از متن و B یک متغیر در آن است به طوری که رشته‌ای مانند w در $L(G)$ وجود دارد که B در تمام اشتقاق‌های ممکن برای آن استفاده می‌شود. ثابت کنید L یک زبان turing-recognizable است.

پاسخ

ماشین تورینگ M که یک recognizer برای این زبان است را می‌سازیم.

ماشین M برای ورودی $< G, B >$ به صورت زیر عمل می‌کند:

۱- گرامر مستقل از متن G\B را با حذف کردن متغیر B از تمام قاعده‌های G می‌سازیم.

۲–یک decider برای A_{CFG} می‌سازیم. (زبان A_{CFG} شامل زوج‌های $< G, w >$ است که در آن G یک گرامر مستقل از متن است که رشته w را تولید می‌کند. تصمیم‌پذیر بودن این زبان در کلاس اثبات شده‌است.)

۳– به ازای هر رشته w که $< G , w > \in A_{CFG}$ است، اگر $< G\backslash B , w > \notin A_{CFG}$ ورودی پذیرفته می‌شود. در غیر این صورت سراغ رشته بعدی می‌رویم.

۴– اگر تمام رشته‌های $L(G)$ را بررسی کردیم و هیچ‌کدام پذیرفته نشد، ورودی را رد می‌کنیم.

زبان $L(G\backslash B)$ فقط شامل رشته‌های $w \in L(G)$ می‌شود که در اشتقاق‌های آن‌ها از B استفاده نمی‌شود. اگر رشته‌ای مانند w وجود داشته باشد که عضو $L(G)$ باشد ولی عضو $L(G\backslash B)$ نباشد می‌توان نتیجه گرفت که B در تمام اشتقاق‌های w وجود دارد و توسط ماشین M پذیرفته می‌شود. اگر چنین رشته‌ای وجود نداشته باشد آن‌گاه $L(G) = L(G\backslash B)$ خواهد بود و در صورتی که زبان $L(G)$ بی‌نهایت باشد ماشین M در حلقه می‌افتد و در غیر این صورت ورودی را رد می‌کند. پس ماشین M یک reocgnizer برای زبان L است.

6) فرض کنید می‌دانیم L_1 یک زبان تصمیم‌پذیر است. اگر زبان L_2 از تمام رشته‌هایی تشکیل شده باشد که پیشوندی از رشته‌های موجود در L_1 هستند، نشان دهید L_1 یک زبان turing-recognizable است.

پاسخ

فرض کنید ماشین تورینگ M زبان L_1 را decide می‌کند. ماشین تورینگ N را به شکل زیر تعریف می‌کنیم:

On input x

1. Let s_1 , s_2 , . . . be all the strings over Σ
2. For i = 1, 2, 3, . . .
   3. Run M on xs_i
   4. If M accepts, then accept. If M rejects, then go to the next i.

ماشین N یا رشته ورودی x را می‌پذیرد یا این که همیشه در loop باقی‌می‌ماند پس می‌توان گفت ماشین N زبان L_2 را recognize می‌کند.

7) (امتیازی) تصمیم پذیری زبان زیر را بررسی کنید.

زبان L روی الفبای {a} تعریف شده و رشته ورودی را به شرطی می‌پذیرد که وقتی تعداد کاراکتر های آن را به شکل باینری نمایش می‌دهیم، تعداد 1 ها فرد باشد. (برای مثال رشته aa پذیرفته می‌شود چون نمایش باینری تعداد کاراکترهای آن 10 است که تعداد فرد ۱ دارد اما aaaaa رد می‌شود چون نمایش باینری تعداد کاراکترهای آن 101 است که تعداد زوج ۱ دارد.)

پاسخ

این زبان تصمیم‌پذیر است. ماشین تورینگ N را به شکل زیر تعرف می‌کنیم. این ماشین زبان L را می‌پذیرد و همواره متوقف می‌شود.

ماشین N باید ابتدا طول رشته ورودی را به نمایش باینری تبدیل کند و سپس، زوجیت تعداد 1 های آن را بررسی کند. برای تبدیل به نمایش باینری، فرض کنید ماشین تورینگ دو نواره باشد(می دانیم ماشین تورینگ چند نواره با ماشین تورینگ استاندارد هم‌ارز است.) اگر رشته اولیه را روی نوار اول داشته باشیم، هر بار از سمت چپ رشته ورودی به سمت راست حرکت کرده و به ازای هر دو a که ببینیم، به جای دومی b می گذاریم. در صورتی که بعد از یک a، blank باشد، تعداد a ها فرد بوده و روی نوار پایینی 1 را می نویسیم و یک خانه چپ می‌رویم. در غیر این‌صورت 0 می‌نویسیم و یک خانه چپ می‌رویم. سپس دوباره head را به سمت چپ رشته ورودی می‌بریم (انقدر چپ می رویم تا به جای a و blank، b ببینیم.) و دوباره این عملیات را انجام می دهیم. (در انجام این عملیات، b ها را نادیده می‌گیریم.) هنگامی کار تمام می‌شود که در یک دور پیمایش رشته، هیچ a ای دیده نشود. در نهایت در نوار دوم، نمایش باینری تعداد a ها ساخته شده است. حال باید یک DFA برای تشخیص زوج یا فرد بودن تعداد ۱ های این رشته بسازیم. دو state برای زوج بودن و فرد بودن تعداد 1 ها در نظر می‌گیریم. از state زوج بودن شروع کرده، روی رشته حرکت کرده و هر بار 1 میبینیم state خود را بین این دو عوض می کنیم. در نهایت وقتی رشته باینری تمام شد، در صورتی که روی state فرد بودیم، رشته را می پذیریم و در غیر این صورت رد می کنیم.

این ماشین زبان L را می‌پذیرد و حتما در انتهای رشته متوقف می‌شود.