به نام خدا

نظریهٔ زبان‌ها و ماشین‌ها

آرش شفیعی

محاسبه‌پذیری

تصمیم‌پذیری

- برای برخی از مسأله‌های محاسباتی هیچ الگوریتمی وجود ندارد. به طور مثال الگوریتمی وجود ندارد که با دریافت یک گرامر مستقل از متن تعیین کند که گرامر مبهم است. به عبارت دیگر این مسأله یک مسأله تصمیم‌ناپذیر است و هیچ ماشین تورینگی وجود ندارد که به ازای همهٔ ورودی‌ها (گرامرهای مستقل از متن) در یک حالت پایانی (در صورتی که گرامر مبهم است) یا یک حالت غیرپایانی (در صورتی که گرامر غیرمبهم است) متوقف شود.
- مفهوم تصمیم‌پذیری را در این قسمت معرفی می‌کنیم و خواهیم دید که چه کارهایی را با ماشین تورینگ نمی‌توان انجام داد. به عبارت دیگر محدودیت ماشین‌ها را در محاسبات بررسی می‌کنیم.
- بسیاری از مسائل تصمیم‌ناپذیر گرچه به آسانی قابل بیان هستند، ولی الگوریتمی برای آنها وجود ندارد.

تصمیم‌پذیری

- یک مسألهٔ تصمیم‌گیری یک عبارت یا گزاره است که پاسخ آن بلی یا خیر است. به عبارت دیگر یک مسأله تصمیم‌گیری، جمله‌ای است که جواب آن بلی است اگر عضو یک زبان تعیین شده باشد و پاسخ آن خیر است اگر عضو آن زبان نباشد.
- برای مثال این مسأله را در نظر بگیرید: به ازای گرامر مستقل از متن G ، آیا G مبهم است؟ پاسخ این سؤال برای برخی از گرامرها بلی است و برای برخی از گرامرها خیر است. دامنهٔ این مسأله، مجموعهٔ همهٔ گرامرهای مستقل از متن است.
- یک مسأله تصمیم‌پذیر است اگر ماشین تورینگی وجود داشته باشد که به ازای همهٔ اعضای دامنه تعیین کند که جواب بلی است (جمله عضو زبان است) یا خیر (جمله عضو زبان نیست).
- دقت کنید که یک مسأله می‌تواند بر روی یک دامنه تصمیم پذیر باشد و بر روی یک دامنهٔ دیگر تصمیم‌پذیر نباشد.

تصمیم‌پذیری

- مسألهٔ دهم هیلبرت را در نظر بگیرید.
- آیا الگوریتمی وجود دارد که به ازای یک چندجمله‌ای داده شده، تعیین کند آیا ریشه‌های چندجمله‌ای اعداد صحیح هستند یا خیر؟
- این مسأله را می‌توانیم به صورت یک مسأله تعیین عضویت یک جمله در یک زبان بنویسیم:
- به ازای چند جمله‌ای p تعیین کنید آیا p عضو زبان D است یا خیر.
  D = {p : p یک چندجمله‌ای با ریشه‌های صحیح است}
- به عبارت دیگر آیا ماشین تورینگی برای زبان D وجود دارد که به ازای همهٔ چندجمله‌ای ها در یک حالت پایانی یا یک حالت غیر پایانی متوقف شود؟

تصمیم‌پذیری

- $D = \{p :$ یک چندجمله‌ای با ریشه‌های صحیح است $p\}$
- برای تشخیص زبان D ماشین تورینگ زیر را در نظر بگیرید:
- ماشین تورینگ M به ازای ورودی p و همهٔ زیر مجموعه‌های مجموعهٔ اعداد صحیح، عبارت p را محاسبه می‌کند. اگر حاصل چندجمله‌ای به ازای مقادیری از اعداد صحیح صفر بود ورودی p پذیرفته می‌شود.
- دقت کنید که اگر p ریشهٔ چندجمله‌ای نداشت ماشین تورینگ M در یک حلقهٔ بی‌پایان می‌افتد، زیرا به جواب نمی‌رسد ولی نمی‌دانیم آیا در نهایت به جواب خواهد رسید یا خیر.
- در اینصورت می‌گوییم ماشین M یک تشخیص‌دهنده [1] است ولی تصمیم‌گیرنده [2] نیست.
- اگر برای دامنهٔ مقادیر ریشه‌های یک چندجمله‌ای در این مسأله یک محدوده تعریف کنیم آنگاه مسأله تصمیم‌پذیر خواهد شد، زیرا در حلقهٔ بی‌پایان نمی‌افتیم.

[1] recognizer
[2] decider

تصمیم‌پذیری

- برای وارد کردن یک مفهوم به ورودی ماشین تورینگ ابتدا باید آن را کدگذاری کنیم. برای مثال اگر بخواهیم گراف G را به ورودی ماشین تورینگ بدهیم، ابتدا گراف را به صورت یک رشته در می‌آوریم. این کدگذاری را با علامت $\langle G \rangle$ نشان می‌دهیم.

تصمیم‌پذیری

- مسأله همبندی گراف را در نظر بگیرید. این مسأله به ازای یک گراف داده شدهٔ G می‌پرسد آیا G همبند است یا خیر.
- حال زبان A را به صورت زیر در نظر بگیرید. $A = \{\langle G \rangle :$ یک گراف همبند است $G\}$
- مسأله همبندی گراف G را به یک مسأله عضویت در زبان G تغییر دادیم. اینک می‌پرسیم به ازای یک رشتهٔ کدگذاری شدهٔ $\langle G \rangle$ برای گراف G آیا $\langle G \rangle$ عضو زبان A است یا خیر.
- برای تشخیص دادن این زبان یک ماشین تورینگ طراحی می‌کنیم که زبان A را بپذیرد. اگر به ازای هر $\langle G \rangle$ ماشین تورینگ رشته را بپذیرد یا رد کند، آنگاه ماشین طراحی شده یک تصمیم‌گیرنده است و زبان A یک زبان تصمیم‌پذیر است.

تصمیم‌پذیری

- زبان A تصمیم‌پذیر است، زیرا الگوریتمی برای آن وجود دارد. این الگوریتم یک توصیف سطح بالا است برای عملکرد یک ماشین تورینگ که A را تصمیم‌می‌گیرد، یا به عبارت دیگر به ازای هر جمله w (که یک گراف کدگذاری شده است) در مجموعهٔ همهٔ گراف‌ها تصمیم می‌گیرد که جمله w عضو زبان A هست یا خیر.

- به ازای گراف G الگوریتم بدین صورت است: (۱) یک رأس را علامت‌گذاری می‌کنیم. (۲) همهٔ همسایه‌های رأس‌های علامت‌گذاری شده را علامت‌گذاری می‌کنیم. (۳) اگر همهٔ رئوس علامت‌گذاری شدند، گراف G همبند است و ⟨G⟩ عضو زبان A است، پس ⟨G⟩ پذیرفته می‌شود، در غیر اینصورت گراف G همبند نیست و ⟨G⟩ عضو زبان A نیست، پس ⟨G⟩ رد می‌شود.

تصمیم‌پذیری

- در مبحث الگوریتم‌ها معمولا یک الگوریتم برای یک مسأله پیدا می‌کنیم و با پیدا کردن یک الگوریتم به طور ضمنی نشان می‌دهیم که مسأله قابل حل است.
- ولی اگر الگوریتمی برای مسأله‌ای پیدا نکنیم، چگونه می‌توانیم با قاطعیت بگوییم که مسأله غیرقابل حل است و الگوریتمی وجود ندارد؟
- در اینجا روشی برای اثبات تصمیم‌ناپذیری ارائه می‌کنیم. اگر ثابت کنیم یک مسأله تصمیم‌ناپذیر است، اثبات کرده‌ایم که الگوریتمی برای آن وجود ندارد.
- اگر بتوانیم اثبات کنیم مسأله‌ای غیرقابل حل است، دیگر به دنبال الگوریتم برای آن نمی‌گردیم.

تصمیم‌پذیری

- مسألهٔ پذیرفتن یک رشته توسط یک ماشین متناهی را در نظر بگیرید. به ازای رشتهٔ w آیا رشته توسط ماشین متناهی قطعی B پذیرفته می‌شود؟

- این مسأله را می‌توانیم به صورت یک زبان نمایش دهیم:
$$A_{DFA} = \{\langle B, w\rangle : \text{یک ماشین متناهی قطعی است که رشتهٔ } w \text{ را می‌پذیرد } B\}$$

- این زبان یک زبان تصمیم‌پذیر است، زیرا الگوریتمی وجود دارد که به ازای ماشین B و رشتهٔ w تعیین کند رشته توسط ماشین پذیرفته می‌شود یا خیر. این الگوریتم می‌تواند توسط یک ماشین تورینگ اجرا شود. به عبارت دیگر ماشین تورینگ M وجود دارد که اجرای رشتهٔ w بر روی ماشین B را شبیه‌سازی می‌کند. اگر رشتهٔ w توسط ماشین B پذیرفته شود، ماشین M در یک حالت پایانی متوقف می‌شود و در غیراینصورت ماشین M در یک حالت غیرپایانی متوقف می‌شود.

- پس ماشین M که برای زبان A_{DFA} طراحی شده است، به ازای هر ورودی $\langle B, w\rangle$ متوقف می‌شود و بنابراین زبان A_{DFA} تصمیم‌پذیر است.

تصمیم‌پذیری

- مسألهٔ تولید یک رشته توسط یک گرامر مستقل از متن را در نظر می‌گیریم. به ازای رشتهٔ w آیا رشته توسط گرامر مستقل از متن G تولید می‌شود؟
- این مسأله را می‌توانیم به صورت یک زبان نمایش دهیم:
$A_{CFG} = \{\langle G, w \rangle :$ یک گرامر مستقل از متن است که رشتهٔ w را تولید می‌کند G$\}$
- آیا زبان A_{CFG} تصمیم‌پذیر است؟

تصمیم‌پذیری

- $A_{CFG} = \{\langle G, w\rangle :$ یک ماشین گرامر مستقل از متن است که رشتهٔ w را تولید می‌کند $G\}$
- زبان A_{CFG} تصمیم‌پذیر است، زیرا یک الگوریتم وجود دارد که تعیین کند آیا رشته عضو زبان است یا خیر.
- گرامر G را به فرم نرمال چامسکی تبدیل می‌کنیم. همهٔ اشتقاق‌ها را با طول $|w|$ به دست می‌آوریم. اگر رشتهٔ w به دست آمد رشته عضو زبان است، در غیراینصورت رشته عضو زبان نیست.

تصمیم‌پذیری

- زبان زیر را در نظر بگیرید: $\{A$ یک ماشین متناهی قطعی است به طوری که $E_{DFA} = \{\langle A\rangle : L(A) = \emptyset$
- آیا این زبان یک زبان تصمیم‌پذیر است؟ به عبارت دیگر آیا الگوریتمی وجود دارد که به ازای یک ماشین متناهی قطعی داده شده، تعیین کند آیا زبان ماشین تهی است یا خیر؟
- و باز به عبارتی آیا ماشین تورینگی وجود دارد که به ازای هر ماشین متناهی قطعی داده شده (به صورت کدگذاری شده) در حالت پایانی متوقف شود اگر زبان ماشین تهی است و در حالت غیرپایانی متوقف شود اگر زبان ماشین غیرتهی است؟
- این زبان یک زبان تصمیم‌پذیر است، زیرا الگوریتمی برای آن وجود دارد. الگوریتم بدین صورت است که اگر حداقل یک مسیر غیرتهی از حالت آغازی ماشین متناهی A به یک حالت پایانی وجود داشته باشد، آنگاه زبان ماشین A غیرتهی است.

تصمیم‌پذیری

- زبان زیر را در نظر بگیرید: $E_{CFG} = \{\langle G\rangle : L(G) = \emptyset$ یک گرامر مستقل از متن است به طوری که $G\}$
- این زبان نیز یک زبان تصمیم‌پذیر است زیر الگوریتمی برای آن وجود دارد.
- ابتدا همهٔ نمادهای پایانی را علامت‌گذاری می‌کنیم. سپس متغیر A که قانونی به صورت $A \to U_1 U_2 \cdots U_n$ دارد را علامت‌گذاری می‌کنیم، اگر همهٔ نمادهای (پایانی یا غیرپایانی) U_i علامت‌گذاری شده باشند.
- اگر در نهایت S علامت‌گذاری نشد، ورودی را قبول می‌کنیم و در غیراینصورت آن را رد می‌کنیم.

تصمیم‌پذیری

- آیا زبان A_{LBA} تصمیم‌پذیر است؟
$A_{LBA} = \{\langle M, w\rangle :$ یک ماشین کراندار خطی است که رشتهٔ w را می‌پذیرد $M\}$

- اگر M یک ماشین کراندار خطی با q حالت باشد و الفبای نوار آن g نماد داشته باشد آنگاه به ازای یک ورودی با طول n این ماشین می‌تواند در qng^n پیکربندی مختلف قرار بگیرد، زیرا محتوای نوار می‌تواند g^n حالت مختلف داشته باشد و هد خواندن نوشتن می‌تواند در n مکان مختلف قرار بگیرد.

تصمیم‌پذیری

- آیا زبان A_{LBA} تصمیم‌پذیر است؟
$A_{LBA} = \{\langle M, w\rangle$: M یک ماشین کراندار خطی است که رشتهٔ w را می‌پذیرد$\}$
- می‌توانیم الگوریتمی برای پذیرفتن یک رشته توسط یک ماشین کراندار خطی بدین صورت طراحی کنیم:

۱. اجرای ماشین M را بر روی رشتهٔ w شبیه‌سازی می‌کنیم. از آنجایی که تعداد کل پیکربندی‌های ممکن qng^n است، شبیه‌سازی را تا qng^n گام ادامه می‌دهیم.

۲. اگر ماشین قبل از اتمام شبیه‌سازی متوقف شد و رشته را پذیرفت ورودی $\langle M, w\rangle$ را می‌پذیریم. اگر ماشین M رشته را نپذیرفت و یا تعداد گام‌های شبیه‌سازی پایان یافت ورودی $\langle M, w\rangle$ را رد می‌کنیم.

- الگوریتمی برای تصمیم‌گیری این زبان وجود دارد، پس این زبان تصمیم‌پذیر است.

تصمیم‌ناپذیری

- حال مسألهٔ زیر را در نظر بگیرید: به ازای ماشین تورینگ M و رشتهٔ w تعیین کنید آیا رشته توسط ماشین پذیرفته می‌شود یا خیر.
- به عبارت دیگر $A_{\mathrm{TM}} = \{\langle M, w\rangle$: M یک ماشین تورینگ است که رشتهٔ w را می‌پذیرد$\}$
- آیا زبان A_{TM} تصمیم‌پذیر است؟

تصمیم‌ناپذیری

- فرض می‌کنیم زبان A_{TM} تصمیم‌پذیر باشد و به تناقض می‌رسیم. فرض کنیم H یک ماشین تصمیم‌پذیر است که زبان A_{TM} را می‌پذیرد.

- با دریافت ورودی $\langle M, w\rangle$ به طوری که M یک ماشین تورینگ دلخواه و w یک جمله است، در صورتی که ماشین M جملهٔ w را بپذیرد و متوقف شود، آنگاه H ورودی $\langle M, w\rangle$ را می‌پذیرد، در غیر اینصورت H ورودی را نمی‌پذیرد.
  $H(\langle M, w\rangle) = accept;$ اگر M جملهٔ w را بپذیرد
  $H(\langle M, w\rangle) = reject;$ اگر M جملهٔ w را نپذیرد

- حال با استفاده از ماشین تورینگ H ماشین تورینگ D را طراحی می‌کنیم.
  $D(\langle M\rangle) = reject;$ آنگاه $H(\langle M, \langle M\rangle\rangle) = accept;$ اگر
  $D(\langle M\rangle) = accept;$ آنگاه $H(\langle M, \langle M\rangle\rangle) = reject;$ اگر

- به عبارت دیگر:
  $D(\langle M\rangle) = reject;$ اگر M جملهٔ $\langle M\rangle$ را بپذیرد
  $D(\langle M\rangle) = accept;$ اگر M جملهٔ $\langle M\rangle$ را نپذیرد

تصمیم‌ناپذیری

- حال اگر به ماشین تورینگ D ورودی $\langle \mathrm{D} \rangle$ را بدهیم، چه اتفاقی می‌افتد؟
- اگر $\mathrm{D}(\langle \mathrm{D} \rangle) = \mathrm{accept}$ آنگاه $\mathrm{H}(\langle \mathrm{D}, \langle \mathrm{D} \rangle \rangle) = \mathrm{reject}$ آنگاه $\mathrm{D}(\langle \mathrm{D} \rangle) = \mathrm{reject}$
- اگر $\mathrm{D}(\langle \mathrm{D} \rangle) = \mathrm{reject}$ آنگاه $\mathrm{H}(\langle \mathrm{D}, \langle \mathrm{D} \rangle \rangle) = \mathrm{accept}$ آنگاه $\mathrm{D}(\langle \mathrm{D} \rangle) = \mathrm{accept}$
- پس در هر صورت به تناقض می‌رسیم و بنابراین فرض اولیه مبنی بر وجود ماشین تورینگ H نادرست است و چنین ماشینی وجود ندارد.

کاهش

- کاهش [1] روشی است برای تبدیل یک مسأله به یک مسألهٔ دیگر به طوری که راه حل مسألهٔ دوم بتواند برای مسألهٔ اول مورد استفاده قرار بگیرد.
- پس وقتی مسألهٔ A را به مسألهٔ B کاهش دهیم جواب مسألهٔ B را می‌توانیم برای مسألهٔ A نیز استفاده کنیم.
- برای مثال در ریاضی مسألهٔ حل کردن یک دستگاه معادلهٔ چند مجهولی را به پیدا کردن وارون یک ماتریس کاهش می‌دهیم.

[1] reduction

کاهش

- در نظریهٔ محاسبات، اگر مسألهٔ A قابل کاهش[1] به مسألهٔ B باشد و B تصمیم‌پذیر باشد، آنگاه A نیز تصمیم‌پذیر است.
- مسألهٔ A را به B کاهش می‌دهیم. اگر به ازای یک ورودی B ورودی را پذیرفت، A نیز ورودی را می‌پذیرد، در غیراینصورت ورودی را نمی‌پذیرد.
- اگر A تصمیم‌ناپذیر باشد و قابل کاهش به B باشد، آنگاه B نیز تصمیم‌ناپذیر است. (فرض کنید B تصمیم‌پذیر باشد، آنگاه الگوریتمی تصمیم‌پذیر برای آن وجود دارد. مسأله A را به B کاهش می‌دهیم و الگوریتمی تصمیم‌پذیر برای A با استفاده از الگوریتم B پیدا می‌کنیم. ولی می‌دانیم A تصمیم‌ناپذیر است، پس B نمی‌تواند تصمیم‌پذیر باشد.)

[1] reducable

کاهش

- حال مسألهٔ توقف [1] را در نظر بگیرید.
- مسألهٔ توقف می‌پرسد آیا ماشین تورینگ M بر روی رشتهٔ w توقف می‌کند یا خیر؟
- $HALT_{TM} = \{\langle M, w\rangle$: M یک ماشین تورینگ است و بر روی رشتهٔ w توقف می‌کند$\}$
- دقت کنید که می‌دانیم $HALT_{TM}$ تشخیص‌پذیر است زیرا برای آن ماشین تورینگی وجود دارد که آن را می‌پذیرد. در اینجا ثابت می‌کنیم که این زبان تصمیم‌پذیر نیست.

[1] halting problem

کاهش

- فرض کنیم مسأله توقف تصمیم‌پذیر باشد. پس ماشین تورینگ R باید وجود داشته باشد که $\mathrm{HALT_{TM}}$ را تصمیم می‌گیرد.

- با استفاده از R ماشین S را می‌سازیم و نشان می‌دهیم که S یک ماشین تصمیم‌پذیر برای مسألهٔ پذیرش $\mathrm{A_{TM}}$ است. از آنجایی که مسألهٔ $\mathrm{A_{TM}}$ تصمیم‌ناپذیر است فرض اولیه نادرست بوده و مسألهٔ توقف نمی تواند تصمیم‌پذیر باشد.

- ماشین S را بدین صورت می‌سازیم: ماشین تورینگ R را روی ورودی $\langle M, w \rangle$ اجرا می‌کنیم. اگر R ورودی را رد، کرد S نیز رد می‌کند. اگر R ورودی را پذیرفت، آنگاه اجرای ماشین M را با ورودی w شبیه‌سازی می‌کنیم. اگر M رشتهٔ w را پذیرفت S ورودی را می‌پذیرد درغیراینصورت S ورودی را رد می‌کند

- پس فرض کردیم $\mathrm{HALT_{TM}}$ تصمیم‌پذیر است و بدین نتیجه رسیدیم که $\mathrm{A_{TM}}$ نیز تصمیم‌پذیر است. ولی قبلا ثابت کردیم که $\mathrm{A_{TM}}$ تصمیم‌ناپذیر است، پس $\mathrm{HALT_{TM}}$ تصمیم‌ناپذیر است.

تصمیم‌ناپذیری

- آنچه به طور رسمی با استدلال منطقی نشان دادیم را می‌توانیم به طور غیررسمی به زبان برنامه نویسی بیان کنیم:
- فرض کنید تابعی به نام `halt` وجود دارد که به ازای هر تابع داده شدهٔ `f` تعیین می‌کند آیا تابع `f` متوقف می‌شود یا خیر.
- حال تابع `f` را بدین صورت در نظر بگیرید:

```
void f() { if (halt(&f)) loop_forever(); }
```

- اگر `halt` مقدار درست را بازگرداند آنگاه تابع `f` باید متوقف شود ولی متوقف نمی‌شود. اگر `halt` مقدار نادرست را بازگرداند آنگاه تابع `f` نباید متوقف شود ولی متوقف می‌شود. پس فرض اولیه مبنی بر وجود تابع `halt` نادرست بوده و چنین تابعی وجود ندارد.

کاهش

- ثابت کنید مسألهٔ $E_{TM} = \{\langle M \rangle : L(M) = \emptyset$ و M یک ماشین تورینگ است$\}$ تصمیم‌ناپذیر است.
- فرض کنید ماشین تورینگ R زبان E_{TM} را تصمیم می‌گیرد. با استفاده از ماشین R ماشین S را می‌سازیم که زبان A_{TM} را تصمیم می‌گیرد. در نتیجه به تناقض می‌رسیم.
- به ازای ورودی M و w در ماشین S چنین تصمیم می‌گیریم:

۱. ماشین تورینگ M_1 را طوری می‌سازیم که به ازای ورودی x اگر $x \neq w$ آنگاه، رشته را رد می‌کند و اگر $x = w$ آنگاه x را می‌پذیرد اگر M رشتهٔ w را بپذیرد و درغیراینصورت x را رد می‌کند.

۲. ماشین R را با ورودی M_1 اجرا می‌کنیم.

۳. اگر R ورودی را بپذیرد، ماشین S ورودی را رد می‌کند و اگر R ورودی را رد کند، ماشین S ورودی را می‌پذیرد.

کاهش

- دقت کنید که در این اثبات ماشین S در هر بار اجرا باید بتواند ماشین $M_۱$ را با استفاده از توصیف ماشین M و رشتهٔ w بسازد. این کار همیشه ممکن است، زیرا $M_۱$ یک کپی از M است با این تفاوت که یک قسمت برای تست $x = w$ دارد.
- پس فرض کردیم ماشین R یک تصمیم‌گیرنده برای زبان E_{TM} است و یک تصمیم‌گیرندهٔ S برای A_{TM} ساختیم. از آنجایی که قبلا اثبات کردیم هیچ تصمیم‌گیرنده‌ای برای A_{TM} وجود ندارد پس به تناقض می‌رسیم و بنابراین فرض اولیه مبنی بر تصمیم‌پذیر بودن E_{TM} نادرست بوده است و E_{TM} تصمیم‌ناپذیر است.

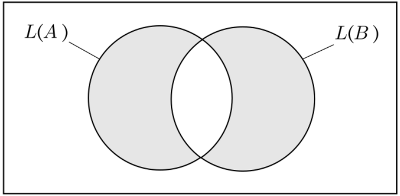
کاهش

- ثابت کنید زبان $\{A$ و B دو ماشین متناهی قطعی هستند و $\mathrm{EQ}_{\mathrm{DFA}} = \{\langle A, B\rangle : L(A) = L(B)$ تصمیم‌پذیر است.

- باید ماشین تورینگی برای این زبان پیدا کنیم که این زبان را بپذیرد.

- با استفاده از زبان A و B زبان C را می‌سازیم به طوری که
$$L(C) = (L(A) \cap \overline{L(B)}) \cup (\overline{L(A)} \cap L(B))$$

- اگر $L(C) = \emptyset$ آنگاه $L(A) = L(B)$



کاهش

- می‌دانیم زبان $E_{DFA} = \{\langle A \rangle : L(A) = \emptyset$ و A یک ماشین متناهی قطعی است$\}$ یک زبان تصمیم‌پذیر است، پس برای آن یک ماشین تورینگ تصمیم‌گیرندهٔ T وجود دارد.
- حال ماشین F را می‌سازیم به طوری که با دریافت دو ماشین متناهی قطعی A و B :

۱. ماشین متناهی قطعی C را می‌سازد به طوری که $L(C) = (L(A) \cap \overline{L(B)}) \cup (\overline{L(A)} \cap L(B))$

۲. با استفاده از ماشین T که یک تصمیم‌گیرنده برای زبان E_{DFA} تصمیم می‌گیریم که آیا زبان $L(C)$ تهی است یا خیر.

۳. اگر T ورودی $\langle C \rangle$ را بپذیرد آنگاه ماشین F ورودی $\langle A, B \rangle$ را قبول می‌کند و اگر T ورودی $\langle C \rangle$ را نپذیرد، آنگاه F نیز ورودی را رد می‌کند.

کاهش

- توجه کنید که استدلال قبل را به دو صورت می‌توانیم تعبیر کنیم:
- (۱) برای اینکه نشان دهیم $\mathrm{EQ_{DFA}}$ تصمیم پذیر است، کافی است یک الگوریتم تصمیم‌پذیر (یک ماشین تورینگ تصمیم‌گیرنده) برای آن پیدا کنیم که در استدلال قبل چنین الگوریتمی یافتیم.
- (۲) برای اینکه نشان دهیم $\mathrm{EQ_{DFA}}$ تصمیم‌پذیر است، از ماشین تصمیم‌گیرنده برای زبان $\mathrm{E_{DFA}}$ استفاده کردیم، پس مسأله را به یک مسأله تصمیم‌پذیر کاهش دادیم.

کاهش

- ثابت کنید زبان $EQ_{TM} = \{\langle M_1, M_2 \rangle : L(M_1) = L(M_2)$ و M_1 و M_2 دو ماشین تورینگ هستند $\}$ تصمیم‌ناپذیر است.

- فرض می‌کنیم زبان EQ_{TM} تصمیم‌پذیر باشد، آنگاه نشان می‌دهیم که در آن صورت زبان E_{TM} باید تصمیم‌پذیر باشد. می‌دانیم که E_{TM} تصمیم‌ناپذیر است، پس فرض اولیه نادرست بوده و EQ_{TM} تصمیم‌ناپذیر است.

کاهش

- ثابت کنید زبان $EQ_{TM} = \{\langle M_۱, M_۲\rangle : L(M_۱) = L(M_۲)$ و $M_۱$ و $M_۲$ دو ماشین تورینگ هستند $\}$ تصمیم‌ناپذیر است.

- فرض می‌کنیم زبان EQ_{TM} تصمیم‌پذیر باشد، آنگاه ماشین تورینگ تصمیم‌گیرندهٔ R برای آن وجود دارد. با استفاده از ماشین R ماشین S را برای E_{TM} می‌سازیم.

- به ازای ورودی $\langle M\rangle$ (رشتهٔ متناظر با ماشین تورینگ M) در ماشین S برای زبان E_{TM} چنین می‌کنیم:

۱. ماشین R را با ورودی $\langle M, M_۱\rangle$ اجرا می‌کنیم به طوری که $M_۱$ ماشینی است که همهٔ ورودی‌ها را رد می‌کند.

۲. اگر ماشین R ورودی $\langle M, M_۱\rangle$ را پذیرفت، این بدین معناست که زبان ماشین M تهی است، پس ورودی $\langle M\rangle$ در ماشین S پذیرفته می‌شود و در غیراینصورت ورودی رد می‌شود.

- اگر R تصمیم‌گیرنده‌ای برای زبان EQ_{TM} باشد، آنگاه S تصمیم‌گیرنده‌ای برای زبان E_{TM} است. اما قبلا نشان دادیم E_{TM} تصمیم‌ناپذیر است، پس فرض اولیه نادرست بوده و EQ_{TM} باید تصمیم‌ناپذیر باشد.

کاهش

- ثابت کنید زبان $\mathrm{ALL}_{\mathrm{DFA}} = \{\langle A \rangle : L(A) = \Sigma^*$ یک ماشین متناهی قطعی است و $A\}$ تصمیم‌پذیر است.
- با استفاده از ماشین تصمیم‌گیرندهٔ T برای زبان $\mathrm{EQ}_{\mathrm{DFA}}$ یک ماشین تصمیم‌گیرنده برای زبان $\mathrm{ALL}_{\mathrm{DFA}}$ می‌سازیم.
- یک ماشین متناهی قطعی B با یک حالت آغازی و پایانی q_0 می‌سازیم. به ازای هر $a \in \Sigma$ گذار $\delta(q_0, a) = q_0$ را تعریف می‌کنیم. این ماشین همهٔ رشته‌های Σ^* را می‌پذیرد.
- حال از ماشین تصمیم‌گیرندهٔ T استفاده می‌کنیم و ماشین‌های قطعی A و B را به عنوان ورودی به ماشین T می‌دهیم. اگر T ورودی را پذیرفت ماشین A پذیرفته می‌شود و درغیراینصورت پذیرفته نمی‌شود.

مسألهٔ تناظر پست

- تا اینجا محاسبه‌پذیری را برای مسائلی در حوزهٔ نظریهٔ زبان‌ها و ماشین‌ها بررسی کردیم. انواع دیگری از مسأله‌های محاسباتی وجود دارند که می‌توان تصمیم‌پذیری و تصمیم‌ناپذیری آنها را با استفاده از روش‌های ذکر شده اثبات کرد.
- مسألهٔ تناظر پست [1] نوعی پازل است. این مسأله، یک مسألهٔ تصمیم‌ناپذیر است.

[1] Post Correspondence Problem (PCP)

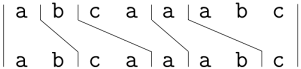
مسألهٔ تناظر پست

- یک دومینو [1] تشکیل شده است از یک رشتهٔ بالایی x و یک رشتهٔ پایینی y که آن را به صورت $\left[\frac{x}{y}\right]$ نشان می‌دهیم.
- یک مجموعه از دومینوها دارای یک تطابق [1] است اگر دنباله‌ای از دومینوها وجود داشته باشد به طوری که الحاق رشته‌های بالایی دومینوها در دنباله برابر با الحاق رشته‌های پایینی دومینوها باشد.

[1] domino
[1] match

مسألهٔ تناظر پست

- برای مثال مجموعهٔ P را در نظر بگیرید: $P = \left\{ \left[\frac{a}{ab}\right], \left[\frac{b}{ca}\right], \left[\frac{ca}{a}\right], \left[\frac{abc}{c}\right] \right\}$
- به ازای اعضای مجموعهٔ P دنبالهٔ دومینوهای زیر دارای یک تطابق است: $\left[\frac{a}{ab}\right]\left[\frac{b}{ca}\right]\left[\frac{ca}{a}\right]\left[\frac{a}{ab}\right]\left[\frac{abc}{c}\right]$
- این تطابق را می‌توانیم به صورت زیر نشان دهیم:



- توجه کنید که اعضای مجموعهٔ P می‌توانند به هر تعداد دلخواه در دنبالهٔ دومینوهای دارای تطابق تکرار شوند. مثلا دومینوی $\left[\frac{a}{ab}\right]$ در دنبالهٔ بالا دو بار تکرار شده است.

مسألهٔ تناظر پست

- برای برخی از مجموعه‌ها تطابقی وجود ندارد. برای مثال در مجموعهٔ $\left\{\left[\frac{\text{abc}}{\text{ab}}\right], \left[\frac{\text{ca}}{\text{a}}\right], \left[\frac{\text{acc}}{\text{ba}}\right]\right\}$ هیچ تطابقی نمی‌توان یافت.
- مسأله تناظر پست می‌پرسد آیا به ازای یک مجموعهٔ داده شدهٔ P یک دنبالهٔ حاوی تطابق وجود دارد یا خیر؟
- دقت کنید یک دومینو می‌تواند به تعداد نامحدودی در یک دنبالهٔ دارای تطابق استفاده شود، پس مجموعهٔ همهٔ پیکربندی‌ها نامحدود است.
- با استفاده از کاهش مسألهٔ A_{TM} به مسألهٔ تناظر پست، می‌توان نشان داد این مسأله تصمیم‌ناپذیر است. به عبارت دیگر برای این مسأله هیچ الگوریتمی وجود ندارد.

مسألهٔ تناظر پست

- یک نمونه از مسألهٔ تناظر پست، یک مجموعهٔ P از دومینوها به صورت $\left\{ \left[\frac{t_۱}{b_۱}\right], \left[\frac{t_۲}{b_۲}\right], \cdots, \left[\frac{t_n}{b_n}\right] \right\}$ است.
- یک تطابق دنباله‌ای به صورت $i_۱, i_۲, \cdots, i_k$ است، به طوری که $t_{i_۱}, t_{i_۲}, \cdots, t_{i_k} = b_{i_۱}, b_{i_۲}, \cdots, b_{i_k}$.
- مسأله این است که آیا مجموعهٔ P دارای یک تطابق است یا خیر.
- می‌توانیم مسألهٔ تطابق پست را بدین صورت بیان کنیم:
- $PCP = \{\langle P \rangle :$ یک نمونه از مسألهٔ تناظر پست است که دارای یک تطابق است $P\}$

مسائل تصمیم‌ناپذیر در مورد زبان‌های مستقل از متن

- حال با استفاده از مسألهٔ تناظر پست، تصمیم‌ناپذیر بودن دو مسأله در مورد زبان‌های مستقل از متن را نشان می‌دهیم.
- فرض کنید P یک نمونه از مسألهٔ تناظر پست به صورت $P = \{(t_1, b_1), (t_2, b_2), \cdots, (t_n, b_n)\}$ است به طوری که t_i رشتهٔ بالایی یک دومینو و b_i رشتهٔ پایینی یک دومینو بر روی الفبای Σ است.
- دو گرامر مستقل از متن G_t و G_b را به طوری می‌سازیم که ویژگی آنها شبیه مسألهٔ تناظر پست باشد.

مسائل تصمیم‌ناپذیر در مورد زبان‌های مستقل از متن

- فرض کنید نمادهای $c_۱, c_۲, \cdots, c_n$ در الفبای Σ نباشند.
- گرامر G_t را با متغیر آغازی S_t و قوانین تولید $(۱ \leq i \leq n)\ S_t \to t_iS_tc_i|t_ic_i$ می‌سازیم.
- به همین ترتیب گرامر G_b را با متغیر آغازی S_b و قوانین تولید $(۱ \leq i \leq n)\ S_b \to b_iS_bc_i|b_ic_i$ می‌سازیم.
- به ازای هر رشتهٔ $c_{i_۱}c_{i_۲}\cdots c_{i_k}$ یک رشتهٔ x در زبان $L(G_t)$ و یک رشتهٔ y در زبان $L(G_b)$ وجود دارد، به طوری که $x = t_{i_k}\cdots t_{i_۲}t_{i_۱}c_{i_۱}c_{i_۲}\cdots c_{i_k}$ و $y = b_{i_k}\cdots b_{i_۲}b_{i_۱}c_{i_۱}c_{i_۲}\cdots c_{i_k}$

مسائل تصمیم‌ناپذیر در مورد زبان‌های مستقل از متن

- حال نشان می‌دهیم دو مسألهٔ زیر تصمیم‌ناپذیرند.

- $\mathrm{EMI_{CFG}}$: به ازای دو گرامر مستقل از متن داده شدهٔ $G_۱$ و $G_۲$ آیا $L(G_۱) \cap L(G_۲) \neq \emptyset$ ؟

$\mathrm{EMI_{CFG}} = \{\langle G_۱, G_۲\rangle :$ $G_۱$ و $G_۲$ دو گرامر مستقل از متن هستند و $L(G_۱) \cap L(G_۲) \neq \emptyset\}$

- $\mathrm{AMB_{CFG}}$: به ازای گرامر مستقل از متن داده شدهٔ G آیا G مبهم است؟

$\mathrm{AMB_{CFG}} = \{\langle G\rangle :$ G یک گرامر مستقل از متن مبهم است$\}$

مسائل تصمیم‌ناپذیر در مورد زبان‌های مستقل از متن

- مسألهٔ تناظر پست PCP را به دو مسألهٔ $\mathrm{EMI}_{\mathrm{CFG}}$ و $\mathrm{AMB}_{\mathrm{CFG}}$ کاهش می‌دهیم.
- فرض کنید I یک نمونه از مسألهٔ تناظر پست باشد به طوری که جواب مسألهٔ I بلی باشد. به عبارت دیگر $\mathrm{I} \in \mathrm{PCP}$.
- آنگاه دنبالهٔ $i_۱, i_۲, \cdots, i_k$ وجود دارد به طوری که $b_{i_k} b_{i_{k-۱}} \cdots b_{i_۱} = t_{i_k} t_{i_{k-۱}} \cdots t_{i_۱}$. بنابراین داریم: $x = t_{i_k} t_{i_{k-۱}} \cdots t_{i_۱} c_{i_۱} \cdots c_{i_k} = b_{i_k} b_{i_{k-۱}} \cdots b_{i_۱} c_{i_۱} \cdots c_{i_k}$.

مسائل تصمیم‌ناپذیر در مورد زبان‌های مستقل از متن

- حال گرامر G را در نظر بگیرید به طوری که $L(G) = L(G_t) \cup L(G_b)$ بنابراین اگر S متغیر آغازی G باشد، داریم: $S \to S_t | S_b$.

- رشتهٔ x هم از گرامر G_t به دست می‌آید و هم از گرامر G_b و بنابراین دو اشتقاق دارد که یکی با $S \Rightarrow S_t$ آغاز می‌شود و دیگری با $S \Rightarrow S_b$. بنابراین G یک گرامر مبهم است. پس نشان دادیم به ازای هر نمونه از مسألهٔ PCP می‌توان گرامری ساخت که مبهم است.

- مسألهٔ PCP را به مسألهٔ AMB_{CFG} کاهش دادیم و می‌دانیم PCP تصمیم‌ناپذیر است، پس AMB_{CFG} تصمیم‌ناپذیر است.

- همچنین اگر رشتهٔ x وجود داشته باشد، این بدین معناست که $L(G_1) \cap L(G_2) \neq \emptyset$. پس به ازای هر نمونه از مسألهٔ PCP دو گرامر وجود دارند که اشتراک آنها غیر تهی است. بنابراین می‌توانیم بدین طریق PCP را به EMI_{CFG} کاهش دهیم، پس مسألهٔ EMI_{CFG} نیز تصمیم‌ناپذیر است.

- ثابت کنید زبان
$\mathrm{REGULAR_{TM}} = \{\langle \mathrm{M} \rangle :$ M یک ماشین تورینگ است و $\mathrm{L(M)}$ یک زبان منظم است$\}$ تصمیم‌ناپذیر است.

- فرض می‌کنیم ماشین تورینگ R زبان $\mathrm{REGULAR_{TM}}$ را تصمیم می‌گیرد. با استفاده از R ماشین S را می‌سازیم که زبان $\mathrm{A_{TM}}$ را تصمیم می‌گیرد. از آنجایی که می‌دانیم $\mathrm{A_{TM}}$ تصمیم‌ناپذیر است، پس فرض اولیه نادرست بوده و $\mathrm{REGULAR_{TM}}$ باید تصمیم‌ناپذیر باشد.

به ازای ورودی $\langle M, w\rangle$ در ماشین S به طوری که M یک ماشین تورینگ و w یک رشته است، چنین می‌کنیم:

- ابتدا یک ماشین M_2 می‌سازیم که چنین عمل می‌کند. به ازای ورودی x به ماشین M_2 اگر $x = 0^n1^n$ آنگاه ماشین M_2 رشتهٔ ورودی را می‌پذیرد و اگر x به فرم دیگری بود، آنگاه ماشین M_2 رشتهٔ x را می‌پذیرد اگر و تنها اگر M رشتهٔ w را بپذیرد. دقت کنید که اگر ماشین R ورودی $\langle M_2\rangle$ را بپذیرد، آنگاه زبان ماشین M_2 منظم است و این تنها در صورتی ممکن است که M_2 همهٔ رشته‌های ورودی را بپذیرد، یعنی $L(M_2) = \Sigma^*$. تنها در صورتی M_2 همهٔ رشته‌های ورودی را می‌پذیرد که M رشتهٔ w را بپذیرد. اما اگر ماشین R ورودی $\langle M_2\rangle$ را نپذیرد، آنگاه ماشین M_2 برخی از مقادیر ورودی x را می‌پذیرد و برخی را رد می‌کند. همچنین ماشین M_2 رشته‌های ورودی به شکل 0^n1^n را می‌پذیرد پس زبان آن منظم نیست. این تنها در صورتی ممکن است که M رشتهٔ w را نپذیرد.

- حال ماشین R را بر روی ورودی $\langle M_۲ \rangle$ اجرا می‌کنیم.
- اگر R ورودی $\langle M_۲ \rangle$ را پذیرفت ماشین S ورودی $\langle M, w \rangle$ را می‌پذیرد و اگر R ورودی $\langle M_۲ \rangle$ را نپذیرفت، آنگاه S ورودی $\langle M, w \rangle$ را نمی‌پذیرد.

قضیهٔ رایس

- در حالت کلی، مسألهٔ تعیین کردن هرگونه ویژگی از زبانی که یک ماشین تورینگ M می‌پذیرد، یک مسألهٔ تصمیم‌ناپذیر است.
- بنابراین همهٔ مسائل به شکل زیر تصمیم‌ناپذیر هستند : تعیین کنید زبانی که توسط یک ماشین تورینگ تشخیص داده می‌شود، ویژگی P را دارد.
- ویژگی P در اینجا می‌تواند مستقل از متن بودن زبان، یا حتی متناهی بودن زبان باشد.
- قضیهٔ رایس [1]: فرض کنید P یک ویژگی از زبانی باشد که توسط یک ماشین تورینگ M پذیرفته می‌شود. مسألهٔ تعیین کردن اینکه ماشین تورینگ M ویژگی P را دارد تصمیم‌ناپذیر است.
- بنابراین زبان $P_{TM} = \{\langle M \rangle$: یک ماشین تورینگ است و $L(M)$ ویژگی P را دارد $M\}$ تصمیم‌ناپذیر است.

[1] Rice's theorem

مقایسهٔ زبان‌ها

	منظم	مستقل از متن	حساس به متن	شمارش‌پذیر بازگشتی
بسته‌بودن بر روی:				
الحاق	بله	بله	بله	بله
اجتماع	بله	بله	بله	بله
بستار-ستاره	بله	بله	بله	بله
متمم	بله	خیر	بله	خیر
اشتراک	بله	خیر	بله	بله
اشتراک با منظم	بله	بله	بله	بله
تصمیم‌پذیری:				
عضویت رشته در زبان	بله	بله	بله	خیر
تهی‌بودن زبان	بله	بله	خیر	خیر
محدود بودن زبان	بله	بله	خیر	خیر
مرجع بودن زبان	بله	خیر	خیر	خیر
برابری دو زبان	بله	خیر	خیر	خیر