$(\Sigma^*) \setminus SA$
$SA' = NSA$
$SA' \supseteq NSA$

e SA NSA e' $\leftarrow \Sigma^*$

تعریف ۱ (زبان‌های NSA و SA)[a]

$e(T_0) = NSA$

$\exists T_0$

$e(T_0) \in NSA \rightarrow e(T_0) \notin L(T_0) = NSA \rightarrow e(T_0) \notin NSA$

$e(T_0) \notin NSA$

$$NSA = \{e(T) \mid T \text{ یک ماشین تورینگ است و } e(T) \notin L(T)\}$$

$e(T_0) \in L(T_0)$

$$SA = \{e(T) \mid T \text{ یک ماشین تورینگ است و } e(T) \in L(T)\}$$

(NSA و SA به ترتیب مخفف non-self-accepting و self-accepting هستند.)

L r.e $\Leftrightarrow$ recursive $L' \Leftrightarrow L'$ recursive

[a] The Languages NSA and SA

قضیه ۱

زبان NSA *شمارشی بازگشتی*[a] نیست. زبان SA شمارشی بازگشتی است ولی بازگشتی نیست.

[a] recursively enumerable

اثبات. فرض کنید NSA بازگشتی‌شمارا باشد؛ در این صورت ماشین تورینگ R وجود دارد به‌گونه‌ای که

$$x \in NSA \iff R \text{ ورودی } x \text{ را می‌پذیرد.}$$

رشتهٔ $w = e(R)$ (کدگذاریِ R) را در نظر بگیرید و دو حالت را بررسی کنید.

(i) R، w را می‌پذیرد. در این صورت $w \in NSA$ و طبق تعریفِ NSA، ماشینی که w را کد می‌کند (یعنی خودِ R) نباید w را بپذیرد؛ تناقض.

(ii) R، w را نمی‌پذیرد. آنگاه $w \notin NSA$ و بنابراین، طبق تعریف، ماشینی که w را کد می‌کند (باز هم R) باید w را بپذیرد؛ تناقض.

در هر دو حالت به تناقض می‌رسیم؛ پس NSA نمی‌تواند بازگشتی‌شمارا باشد.

بگذارید E زبان $\{e(T) \mid T$ یک ماشین تورینگ است$\}$ باشد.

برای هر رشته‌ی z شامل صفر و یک، دقیقاً یکی از سه حالت زیر برقرار است:

۱. z نمایشگر هیچ ماشین تورینگی نیست؛

۲. z نمایشگر ماشینی تورینگ است که z را می‌پذیرد؛

۳. z نمایشگر ماشینی تورینگ است که z را نمی‌پذیرد. به عبارت دیگر،

$$\{0,1\}^* = NSA \cup SA \cup E'$$

و این سه مجموعه در سمت راست، نسبت به هم نابه‌هم‌پوشان (ناشتراک) هستند. بنابراین:

$$NSA = (SA \cup E')' = SA' \cap E$$

اگر SA بازگشتی بود، آنگاه SA' نیز بازگشتی می‌بود، طبق قضیه ۶.۸، و در نتیجه NSA نیز بازگشتی می‌شد، طبق قضیه ۵.۸. اما NSA شمارشی بازگشتی نیست، و بنابراین طبق قضیه ۲.۸ نمی‌تواند بازگشتی باشد.

برای پایان دادن به اثبات، باید نشان دهیم که SA شمارشی بازگشتی است. در ادامه الگوریتمی برای پذیرش SA ارائه می‌دهیم. رشته‌ای $x \in \{0,1\}^*$ را دریافت کنید، و بررسی کنید که آیا $x \in E$ هست یا نه. اگر نه، آنگاه $x \notin SA$. اگر $x \in E$، آنگاه $x = e(T)$ برای یک ماشین تورینگ T است، و همان‌طور که در بخش ۸.۷ مشاهده کردیم، می‌توانیم T را از رشته‌ی $e(T)$ بازسازی کنیم. سپس T را روی ورودی x اجرا کنید و اگر T رشته‌ی x را بپذیرد، آنگاه x را بپذیرید. □

تعریف ۲ (مسائل تصمیم‌پذیر[a])

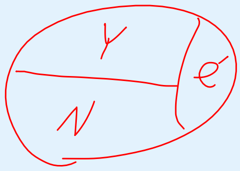
[a]Decidable Problems

اگر P یک *مسئلهٔ تصمیم*[a] باشد و e کدگذاریِ *معقول*[b] نمونه‌های P روی الفبای Σ باشد، می‌گوییم P *تصمیم‌پذیر*[c] است هرگاه

$$Y(P) = \{e(I) \mid \text{یک نمونهٔ بَلی از } P \text{ است } I\}$$

یک زبان *بازگشتی*[d] باشد.

Σ^*



[a]decision problem
[b]reasonable encoding
[c]decidable
[d]recursive language

قضیه ۲

برای هر مسئلهٔ تصمیم P، مسئلهٔ P تصمیم‌پذیر است اگر و تنها اگر مسئلهٔ متمم آن P' نیز تصمیم‌پذیر باشد.

اثبات. داریم

$$N(P) = Y(P)' \cap E(P)$$

فرض شده است که $E(P)$ بازگشتی است. اگر $Y(P)$ بازگشتی باشد، آنگاه $Y(P)'$ نیز بازگشتی است، و بنابراین $N(P) = Y(P')$ نیز بازگشتی است. جهت دیگر نیز مشابه است. □

$Y(P') = N(P)$

$N(P) = (Y(P))' \cap E$

تعریف ۳ (تقلیل یک مسئلهٔ تصمیم یا یک زبان به دیگری)[a]

[a]Reducing One Decision Problem to Another, and Reducing One Language to Another

فرض کنید P_1 و P_2 دو *مسئلهٔ تصمیم*[a] باشند. می‌گوییم P_1 به P_2 *قابل تقلیل*[b] است (می‌نویسیم $P_1 \le P_2$) هرگاه الگوریتمی وجود داشته باشد که برای هر نمونهٔ دلخواه I از P_1، نمونه‌ای $F(I)$ از P_2 بسازد، به‌گونه‌ای که برای هر I، پاسخ دو مسئله یکی باشد؛ یعنی I یک نمونهٔ *بلی* از P_1 باشد اگر و تنها اگر $F(I)$ یک نمونهٔ *بلی* از P_2 باشد.
اگر L_1 و L_2 دو زبان روی الفباهای Σ_1 و Σ_2 باشند، به ترتیب می‌گوییم L_1 به L_2 *قابل تقلیل* است (می‌نویسیم $L_1 \le L_2$) هرگاه تابعی *قابل‌محاسبه با ماشین تورینگ*[c] به صورت

$$f : \Sigma_1^* \to \Sigma_2^*$$

وجود داشته باشد که برای هر رشتهٔ $x \in \Sigma_1^*$، داشته باشیم

$$x \in L_1 \quad \text{اگر و تنها اگر} \quad f(x) \in L_2.$$

$f(y)$

$x \in L_2$ $\qquad y \in L_1$

[a]decision problem
[b]reducible
[c]Turing-computable

قضیه ۳

فرض کنید $L_1 \subseteq \Sigma_1^*$، $L_2 \subseteq \Sigma_2^*$ و $L_1 \le L_2$. اگر L_2 *بازگشتی* باشد، آنگاه L_1 نیز *بازگشتی* است.
همچنین، فرض کنید P_1 و P_2 دو مسئلهٔ تصمیم باشند و $P_1 \le P_2$. اگر P_2 *تصمیم‌پذیر* باشد، آنگاه P_1 نیز تصمیم‌پذیر است.

تعریف ۴

Accepts: اگر یک ماشین تورینگ T و رشته‌ای w داده شده باشند، آیا $w \in L(T)$ است؟
Halts: اگر یک ماشین تورینگ T و رشته‌ای w داده شده باشند، آیا T روی ورودی w متوقف می‌شود؟

اثبات. برای نشان دادن تصمیم‌ناپذیری Accepts، نشان دادیم که:

$$\text{Self-Accepting} \le \text{Accepts}$$

با استفاده از تابع کدگذاری e، اگر $T_1 = T$ و $y = e(T)$، آنگاه زوج (T_1, y) همان ویژگی مورد نیاز را دارد و به‌صورت الگوریتمی از T به‌دست می‌آید.

اکنون برای اثبات تصمیم‌ناپذیری Halts، کافی‌ست نشان دهیم:

$$\text{Accepts} \le \text{Halts}$$

فرض کنید نمونه‌ای (T, x) از Accepts داده شده است. هدف این است که ماشینی T_1 بسازیم که:

$$T_1 \text{ روی } x \text{ متوقف می‌شود} \iff T \text{، رشتهٔ } x \text{ را می‌پذیرد}$$

برای این منظور، ماشینی T_1 می‌سازیم که مانند T عمل می‌کند، با این تفاوت که:

- اگر T بخواهد x را بپذیرد، T_1 نیز آن را می‌پذیرد و متوقف می‌شود.
- اگر T بخواهد x را رد کند، T_1 وارد حلقه‌ی بی‌نهایت می‌شود.

در نتیجه، اگر T رشته‌ی x را بپذیرد، آنگاه T_1 نیز متوقف می‌شود، و در غیر این صورت، متوقف نمی‌شود. بنابراین تابع F به‌صورت $F(T,x)=(T_1,x)$ یک کاهش از Accepts به Halts است. □

قضیه ۴

پنج مسئله‌ی تصمیم‌ناپذیر زیر را در نظر بگیرید:

۱. $Accepts\text{-}\Lambda$: آیا برای یک ماشین تورینگ T، رشته‌ی تهی $\Lambda \in L(T)$ است؟
۲. $AcceptsEverything$: آیا برای یک ماشین تورینگ T با الفبای ورودی Σ، داریم $L(T)=\Sigma^*$؟
۳. $Subset$: آیا برای دو ماشین تورینگ T_1 و T_2 داریم $L(T_1)\subseteq L(T_2)$؟
۴. $Equivalent$: آیا برای دو ماشین تورینگ T_1 و T_2 داریم $L(T_1)=L(T_2)$؟
۵. $WritesSymbol$: آیا برای یک ماشین تورینگ T و یک نماد a از الفبای نوار T، ماشین T در صورتی که نوار ورودی ابتدا تهی باشد، هرگز نماد a را روی نوار می‌نویسد؟

اثبات. در هر مورد، با کاهش از مسئله‌ای تصمیم‌ناپذیر به مسئلهٔ مورد نظر، تصمیم‌ناپذیری آن را نشان می‌دهیم.

۱. $Accepts \leq Accepts\text{-}\Lambda$: یک نمونه از Accepts زوجی (T,x) است. ماشینی T_1 بسازید که ابتدا x را روی نوار بنویسد و سپس مانند T عمل کند. در این صورت، T رشتهٔ x را می‌پذیرد اگر و تنها اگر T_1 رشتهٔ تهی Λ را بپذیرد.

۲. $Accepts\text{-}\Lambda \leq AcceptsEverything$: ماشینی T_1 تعریف کنید که ابتدا ورودی را پاک کند و سپس مانند T عمل کند. در این صورت، برای هر ورودی، رفتار T_1 مانند رفتار T روی Λ است. پس T رشتهٔ تهی را می‌پذیرد اگر و تنها اگر T_1 همهٔ رشته‌ها را بپذیرد.

۳. $AcceptsEverything \leq Subset$: برای ماشین تورینگ T، ماشینی T_1 تعریف کنید که همهٔ رشته‌ها را بپذیرد (مثلاً در قدم اول وارد حالت پذیرش شود) و بگذارید $T_2=T$. آنگاه داریم: $L(T)=\Sigma^*$ اگر و تنها اگر $L(T_1)=\Sigma^*\subseteq L(T)=L(T_2)$.

۴. $Subset \leq Equivalent$: برای ماشین‌های T_1 و T_2، ماشینی T_3 بسازید که $L(T_3)=L(T_1)\cap L(T_2)$، و بگذارید $T_4=T_1$. آنگاه داریم: $L(T_1)\subseteq L(T_2)$ اگر و تنها اگر $L(T_3)=L(T_4)$.

۵. $Accepts\text{-}\Lambda \leq WritesSymbol$: برای ماشین تورینگ T و نماد نوار σ، ماشینی T_1 بسازید که مانند T عمل می‌کند(الفبای ماشین به نحوی عوض شده است که $\sigma \notin \Sigma$)، ولی هرگاه T به حالت پذیرش می‌رسد، T_1 ابتدا نماد σ را روی نوار بنویسد و سپس متوقف شود. آنگاه T رشتهٔ Λ را می‌پذیرد اگر و تنها اگر T_1 روی نوار تهی، نماد σ را می‌نویسد.

□

تعریف ۵

$WritesNonblank$: آیا برای یک ماشین تورینگ T، هنگام اجرای آن روی ورودی تهی Λ، T هرگز نمادی غیرتهی[a] روی نوار می‌نویسد؟

[a] nonblank

قضیه ۵

مسئلهٔ تصمیم $WritesNonblank$ *تصمیم‌پذیر*[a] است.

[a] decidable

تعریف ۶(ویژگی زبانی ماشین‌های تورینگ)

یک ویژگی R از ماشین‌های تورینگ، *ویژگی زبانی*[a] نامیده می‌شود اگر برای هر ماشین تورینگ که ویژگی R را دارد، و هر ماشین دیگر T_1 که $L(T_1) = L(T)$، آنگاه T_1 نیز دارای ویژگی R باشد.
ویژگی زبانی *ناپیش‌پاافتاده*[b] است اگر دست‌کم یک ماشین تورینگ وجود داشته باشد که آن ویژگی را دارد، و دست‌کم یکی که آن را نداشته باشد.

[a] language property
[b] nontrivial

قضیه ۶(قضیهٔ رایس)[a]

[a] Rice's Theorem

اگر R یک ویژگی زبانی *ناپیش‌پاافتاده*[a] برای ماشین‌های تورینگ باشد، آنگاه مسئلهٔ تصمیم زیر تصمیم‌ناپذیر است:

$$P_R: \text{آیا یک ماشین تورینگ } T \text{ ویژگی } R \text{ را دارد؟}$$

[a] nontrivial

اثبات. نشان می‌دهیم که مسئلهٔ تصمیم‌ناپذیر Accepts-Λ به P_R یا به متمم آن $P_{\text{not-}R}$ قابل کاهش است. طبق قضیهٔ ۲، اگر یکی از این دو مسئله تصمیم‌ناپذیر باشد، دیگری نیز تصمیم‌ناپذیر است. بنابراین برای اثبات قضیه، کافی است یکی از این دو کاهش را ارائه دهیم.

از آنجا که R یک ویژگی زبانی ناپیش‌پاافتاده است، دو ماشین تورینگ T_R و $T_{\text{not-}R}$ وجود دارند که اولی ویژگی R را دارد و دومی آن را ندارد. حتی ممکن است فقط یکی از آن‌ها کافی باشد. الگوریتمی می‌سازیم که برای هر ماشین T، ماشینی جدید $T_1 = F(T)$ تولید کند، به‌گونه‌ای که T رشته‌ی تهی را می‌پذیرد اگر و تنها اگر T_1 ویژگی R را داشته باشد (یا برعکس، بسته به نیاز).

برای ساخت T_1، رفتار آن به صورت زیر است:

- ابتدا سر نوار را به اولین خانهٔ تهی پس از ورودی منتقل می‌کند؛
- سپس مانند ماشین T عمل می‌کند و وانمود می‌کند که ورودی‌اش Λ بوده؛
- اگر T رد کند، T_1 نیز رد می‌کند؛
- اگر T بپذیرد، آنگاه T_1 محتوای نوار را پاک کرده و به وضعیت اولیه برمی‌گردد و سپس مانند T_R روی ورودی اصلی اجرا می‌شود.

اگر T رشته‌ی Λ را بپذیرد، آنگاه T_1 زبان $L(T_R)$ را می‌پذیرد و بنابراین ویژگی R را دارد.

اگر T رشته‌ی Λ را نپذیرد (یعنی رد کند یا هرگز متوقف نشود)، آنگاه T_1 مانند یک ماشین ساده‌ی T_0 عمل می‌کند که هیچ رشته‌ای

را نمی‌پذیرد. اگر T_0 ویژگی R را نداشته باشد، در این صورت ما کاهشی از Accepts-Λ به P_R داریم. اگر T_0 ویژگی R را داشته باشد، می‌توانیم از $T_{\text{not-}R}$ در جای T_R استفاده کنیم و نتیجه بگیریم که T_1 ویژگی R را ندارد اگر و تنها اگر T رشته‌ی تهی را بپذیرد. بنابراین در این حالت کاهش به $P_{\text{not-}R}$ داریم.

در هر دو حالت، یکی از مسائل P_R یا $P_{\text{not-}R}$ تصمیم‌ناپذیر است، پس حکم قضیه برقرار است. □

تعریف ۷ (Post's Correspondence Problem)

یک نمونه از مسئلهٔ تطابق Post (یا PCP) مجموعه‌ای از زوج‌ها به‌صورت

$$\{(\alpha_1, \beta_1), (\alpha_2, \beta_2), \ldots, (\alpha_n, \beta_n)\}$$

است که در آن $n \geq 1$ و تمام رشته‌های α_i و β_i ناتهی‌اند و روی یک الفبای ثابت Σ تعریف شده‌اند.
مسئلهٔ تصمیم: آیا عدد صحیح مثبتی k و دنباله‌ای از اندیس‌ها $i_1, i_2, \ldots, i_k$ با $1 \leq i_j \leq n$ وجود دارد به‌گونه‌ای که

$$\alpha_{i_1}\alpha_{i_2}\ldots\alpha_{i_k} = \beta_{i_1}\beta_{i_2}\ldots\beta_{i_k}\ ?$$

یک نمونه از MPCP، همانند PCP است، با این تفاوت که در آن دنبالهٔ اندیس‌ها الزاماً باید با 1 آغاز شود.
یعنی می‌پرسیم: آیا عدد مثبت k و دنباله‌ای $i_2, i_3, \ldots, i_k$ وجود دارد به‌طوری‌که:

$$\alpha_1\alpha_{i_2}\ldots\alpha_{i_k} = \beta_1\beta_{i_2}\ldots\beta_{i_k}\ ?$$

در هر دو حالت (PCP و MPCP)، به چنین مجموعه‌ای یک *سامانهٔ تطابق*[a] یا *سامانهٔ تطابق اصلاح‌شده*[b] گفته می‌شود.
اگر جواب «بلی» باشد، گفته می‌شود که یک *تطابق*[c] وجود دارد؛ یعنی دنباله‌ای از اندیس‌ها که دو رشتهٔ حاصل را برابر می‌سازد.

[a] correspondence system
[b] modified correspondence system
[c] match

قضیه ۷

MPCP به PCP قابل کاهش است، یعنی:

$$\text{MPCP} \leq \text{PCP}$$

اثبات. فرض کنید یک نمونه از MPCP به صورت زیر داده شده باشد:

$$I = \{(\alpha_1, \beta_1), (\alpha_2, \beta_2), \ldots, (\alpha_n, \beta_n)\}$$

می‌خواهیم از روی این نمونه، نمونه‌ای از PCP بسازیم:

$$J = \{(\alpha'_1, \beta'_1), (\alpha'_2, \beta'_2), \ldots, (\alpha'_n, \beta'_n), (\alpha'_{n+1}, \beta'_{n+1})\}$$

در این ساختار، از نمادهای جدید # و \$ استفاده می‌کنیم. اگر $(\alpha_i, \beta_i) = (a_1a_2\ldots a_r, b_1b_2\ldots b_s)$، آنگاه تعریف می‌کنیم:

$$(\alpha'_i, \beta'_i) = (\,\#a_1\#a_2\#\ldots\#a_r\#,\ \#b_1\#b_2\#\ldots\#b_s\#)$$

و در نهایت:

$$(\alpha'_{n+1}, \beta'_{n+1}) = (\$, \#\$)$$

واضح است که زوج‌های I متناظر با زوج‌های اول تا n از J هستند. اگر رشته‌ی حاصل از $\alpha_1\alpha_{i_2}\dots\alpha_{i_k}$ برابر باشد با $\beta_1\beta_{i_2}\dots\beta_{i_k}$، آنگاه داریم:

$$\alpha'_1\alpha'_{i_2}\dots\alpha'_{i_k}\alpha'_{n+1} = \beta'_1\beta'_{i_2}\dots\beta'_{i_k}\beta'_{n+1}$$

از طرف دیگر، به‌علت نحوه‌ی ساخت J، هر تطابق باید از زوج اول آغاز شود و به زوج $(n+1)$ ختم شود، چون رشته‌ی شروع باید با $ ختم شود و فقط زوج پایانی دارای این ویژگی است.

پس اگر:

$$\alpha'_{i_1}\alpha'_{i_2}\dots\alpha'_{i_k} = \beta'_{i_1}\beta'_{i_2}\dots\beta'_{i_k}$$

برای دنباله‌ای از اندیس‌ها برقرار باشد، آنگاه $i_1 = 1$ و $i_k = n+1$ است. در این صورت دنباله‌ی $1, i_2, \dots, i_{k-1}$ تطابقی برای نمونه‌ی اولیه‌ی I در MPCP است.

در نتیجه، هر تطابق برای J در PCP معادل تطابقی برای I در MPCP است. بنابراین، MPCP به PCP کاهش‌پذیر است. □

قضیه ۸

Accepts به MPCP قابل کاهش است:

$$\text{Accepts} \le \text{MPCP}$$

اثبات. می‌خواهیم برای هر نمونه از Accepts شامل یک ماشین تورینگ T و یک رشته‌ی ورودی w، یک سامانه‌ی تطابق اصلاح‌شده‌ی MPCP بسازیم به‌طوری‌که:

$$T \text{ رشته‌ی } w \text{ را می‌پذیرد} \iff \text{سامانهٔ تطابق متناظر تطابق دارد.}$$

ایده کلی ساخت رشته‌ها و زوج‌ها به‌گونه‌ای ساخته می‌شوند که تطابق‌های جزئی (partial match) با توالی پیکربندی‌های T بر w مطابقت داشته باشند. از نمادهای پیکربندی‌های T به‌علاوهٔ نماد جدید # استفاده می‌کنیم.

چهار ویژگی هدف

۱. اگر T توالی پیکربندی‌های $q_0w, x_1q_1y_1, \dots, x_jq_jy_j$ را طی کند، تطابق جزئی به‌شکل زیر به‌دست می‌آید:

$$\alpha = \#q_0w\#x_1q_1y_1\#\dots\#x_jq_jy_j$$

۲. هر تطابق جزئی یا از این فرم پیروی می‌کند یا بعد از رسیدن به پیکربندی پذیرش xh_ay از آن منحرف می‌شود.

۳. اگر چنین تطابقی به xh_ay ختم شود، زوج‌هایی تعریف می‌کنیم که به α اجازه می‌دهند تا به β برسد و تطابق کامل شکل بگیرد.

۴. تنها زمانی که تطابق کامل ایجاد می‌شود، زمانی است که T رشته‌ی w را می‌پذیرد.

ساخت زوج‌ها

- زوج اول:

$$(\alpha_1, \beta_1) = (\#, \#q_0w\#)$$

- زوج‌های نوع ۱: برای هر نماد $a \in \Gamma \cup \{\Delta\}$:

$$(a, a), \quad (\#, \#)$$

- زوج‌های نوع ۲: برای هر گذار $\delta(q, a) = (p, b, D)$، زوج مناسب با نوع حرکت به‌صورت زیر تعریف می‌شود:
  - $D = R$: (qa, bp)
  - $D = S$: (qa, pb)
  - $D = L$: (cqa, pcb)

  حالت‌های مربوط به حرکت روی نماد تهی نیز مشابه تعریف می‌شوند.

- زوج‌های نوع ۳: برای رسیدن از پیکربندی پذیرش به پایان، مانند:

$$(haa, ha), \quad (aha, ha), \quad (ahab, ha)$$

- زوج نوع ۴:

$$(ha\#\#, \#)$$

درستی کاهش اگر T رشته‌ی w را بپذیرد، تطابق جزئی از پیکربندی‌های متوالی $z_0, z_1, \ldots, z_j$ تشکیل می‌شود. با استفاده از زوج‌های نوع ۳ و ۴، تطابق کامل ساخته می‌شود.

اگر T رشته‌ی w را نپذیرد، آنگاه وارد حلقه‌ی بی‌نهایت می‌شود و هیچ‌گاه به حالت h_a نمی‌رسد؛ بنابراین زوج نوع ۴ استفاده نمی‌شود و چون تنها این زوج تعادل تعداد #ها را برقرار می‌کند، هیچ تطابقی ممکن نیست.

اگر $w = \Lambda$ باشد، تنها تغییر لازم این است که زوج اول $(\#, \#q_0\#)$ تعریف شود.

□

قضیه ۹

مسئلهٔ تطابق Post (PCP) تصمیم‌ناپذیر است.

اثبات. این قضیه نتیجه‌ای مستقیم از قضایای ۸ و ۹ است. چون Accepts به MPCP و سپس MPCP به PCP کاهش‌پذیر است، و Accepts تصمیم‌ناپذیر است، پس PCP نیز تصمیم‌ناپذیر است. □