شهزاد ممیز ، ۸۱۰۱۰۰۲۷۲

تکلیف ۱۱ نظریه زبان ها و ماشین ها

سوال ①. $EQ_{CFG} = \{\langle G_1, G_2\rangle \mid G_1, G_2 \text{ are CFGs } \& \ L(G_1) = L(G_2)\}$

برای حل این سوال ، ابتدا در نظر می گیریم که زبان ALL_{CFG} undecidable است.

حل ALL_{CFG} را به EQ_{CFG} کاهش می دهیم. (عمل reduction را انجام میدیم)

$$ALL_{CFG} \leq EQ_{CFG}$$

می خواهیم ثابت کنیم EQ_{CFG} $\boxed{\text{تصمیم پذیر نیست}}$. با برهان خلف ثابت می کنیم.

<u>فرض خلف</u> : EQ_{CFG} تصمیم پذیر است.

اجازه می دهیم که TM به $\underline{R}$ عضو بودن در EQ_{CFG} را decide کند. حال یک TM به $\underline{S}$ می سازیم که عضو بودن در ALL_{CFG} را decide کند.

در ضمن فرض می کنیم $\{0,1\} \in \Sigma^*$

برای decide کردن ALL_{CFG} :

① گرامر G_{All} را به طوری در نظر می گیریم که یک CFG است.

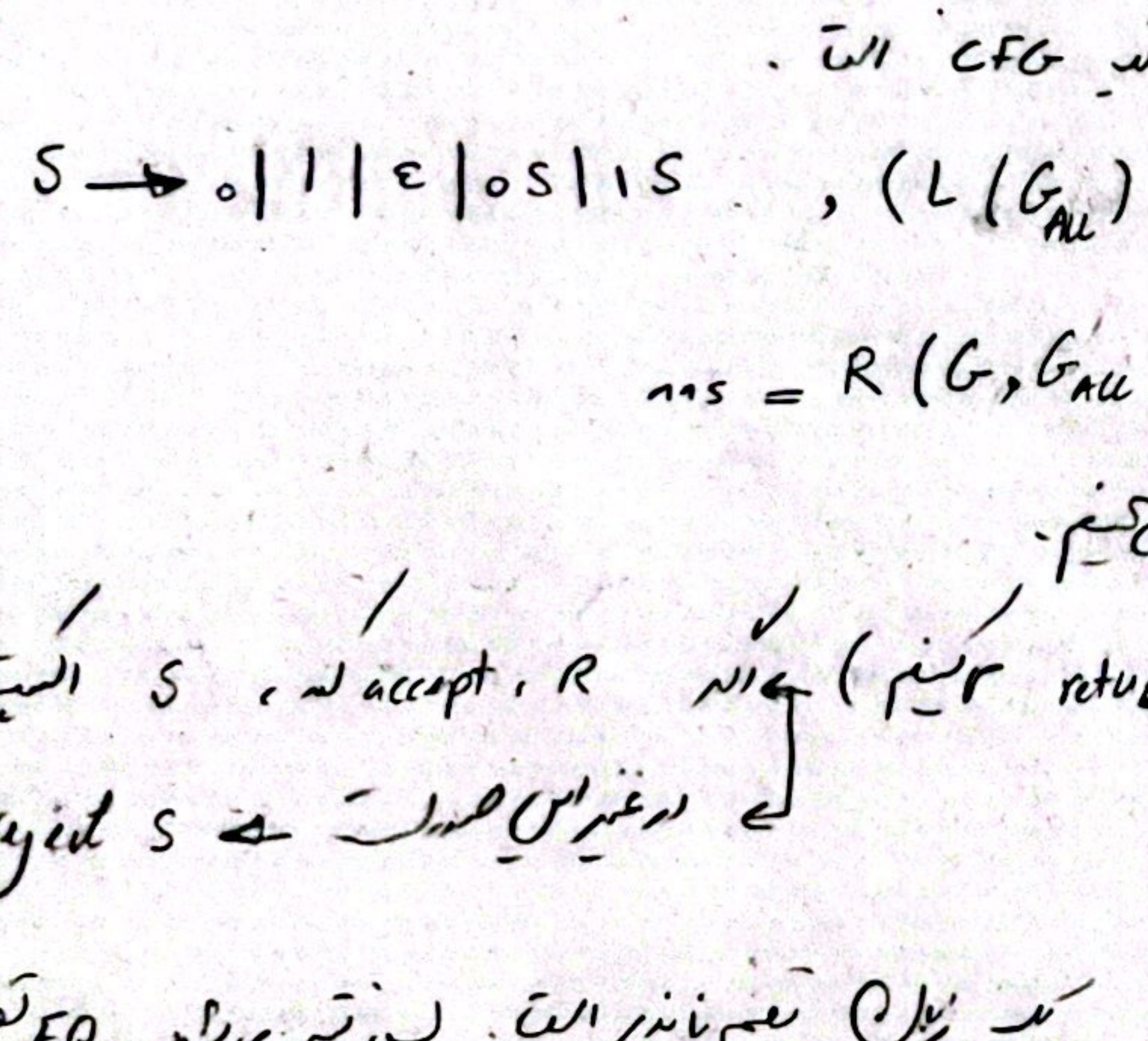
$S \rightarrow 0 \mid 1 \mid \varepsilon \mid 0S \mid 1S \quad , (L(G_{All}) = \Sigma^*)$

② ans را به طور زیر در نظر می گیریم $ans = R(G, G_{All})$

R را برای G و G' ران می کنیم.

③ ans را برمی گردانیم (return می کنیم) ← اگر R ، accept کند ، S اکسپت می کند
← در غیر این صورت S reject می کند

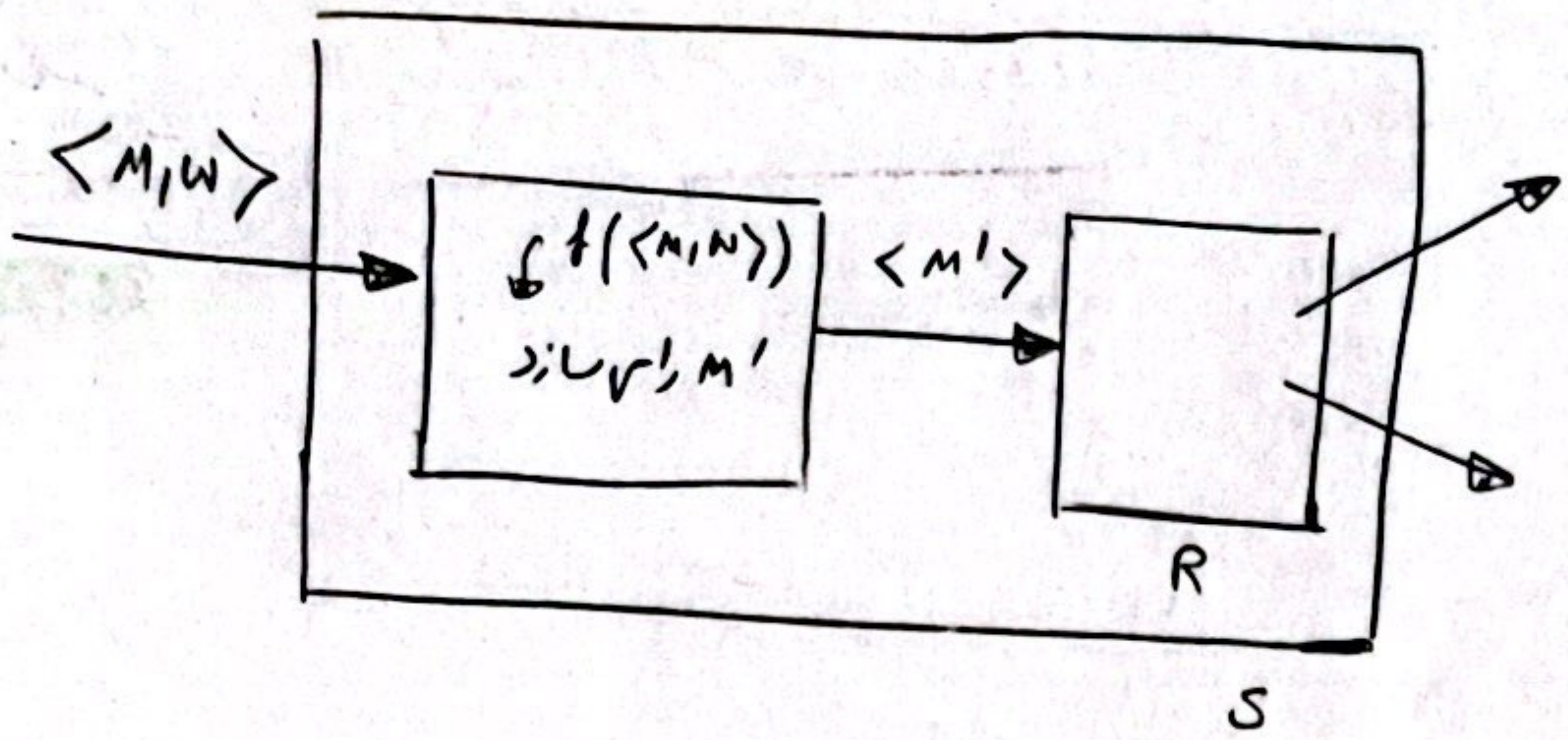
به تناقض خوردیم . زیرا می دانستیم ALL_{CFG} یک زبان تصمیم ناپذیر است . پس نتیجه می گیریم EQ_{CFG} تصمیم ناپذیر است.



اگر $L(G') = L(G)$ باشد ، accept می کند

در غیر این صورت reject می کند.

سوال ۲



accept می‌کند اگر، R روی tape سوم non-blank بنویسد.

در غیر این صورت reject می‌کند.

ماشین تورینگ صورت سوال را $\underline{R}$ می‌نامیم.

$$A_{TM} \leq R_{TM}$$

باید چنین ثابت کنیم از undecidable بودن A_{TM}، به undecidable بودن R_{TM} برسیم.

با برهان خلف ثابت می‌کنیم،

$\underline{S}$ ماشین تورینگ A_{TM} است. f(<M,w>) ورودی <M,w> را می‌گیرد و <M'> را می‌سازد

اگر M ، w را accept کند، که non-blank روی tape دوم می‌نویسد.

حال R را روی M' اجرا می‌کنیم. ← اگر R ، accept کند، یعنی M نیز w را accept کرده
پس یعنی $\underline{S}$ نیز accept می‌کند

← در غیر این صورت $\underline{S}$ reject می‌کند.

که این یعنی TM هم decidable است، اما ما می‌دانیم A_{TM} ، undecidable است. پس به تناقض رسیدیم. ⇐ L_R نیز undecidable است.

سوال ۳.

مسئله PCP ما در اینجا تنها یک عضو دارد. (فرض کنید تنها کاراکتر a را داریم)

در این جا ۲ حالت را در نظر می‌گیریم:

$\boxed{\text{حالت ۱}}$: تعداد حروف قسمت های بالایی دومینو با تعداد حروف قسمت های پایینی برابر نباشد و بلاک A (بالایی) در تمامی دومینو ها کوتاه‌تر یا بلندتر از بلاک B باشد. در این صورت مسئله جواب نمی‌دهد. $n(B) \neq n(A)$

$\boxed{\text{حالت ۲}}$ هر دو بلاک A , B از دومینو، هم سایز باشند $\leftarrow n(B) = n(A)$

در این حالت پاسخ درست را می‌دهیم $\leftarrow$ accept

$\boxed{\text{حالت ۳}}$: این هم مثل حالت ۱ $n(B) \neq n(A)$ است ولی در حالتی که بعضی دومینو ها در بلاک اول (A) بلندتر و در بقیه بلاک دوم (B) بلندتر است. (البته می‌توانند هم $n(A) = n(B)$ هم باشد)

فرض کنید در دومینویی طول بلاک A به اندازه y حرف طولانی‌تر باشد. (این دومینو را Y می‌نامیم)

و در دومینویی طول B به اندازه x حرف طولانی‌تر باشد (اسم این دومینو رو X می‌نامیم)

حال y تا دومینوی X , x تا دومینوی Y را کنار هم می‌چینیم و این خودش یک پاسخ می‌باشد.

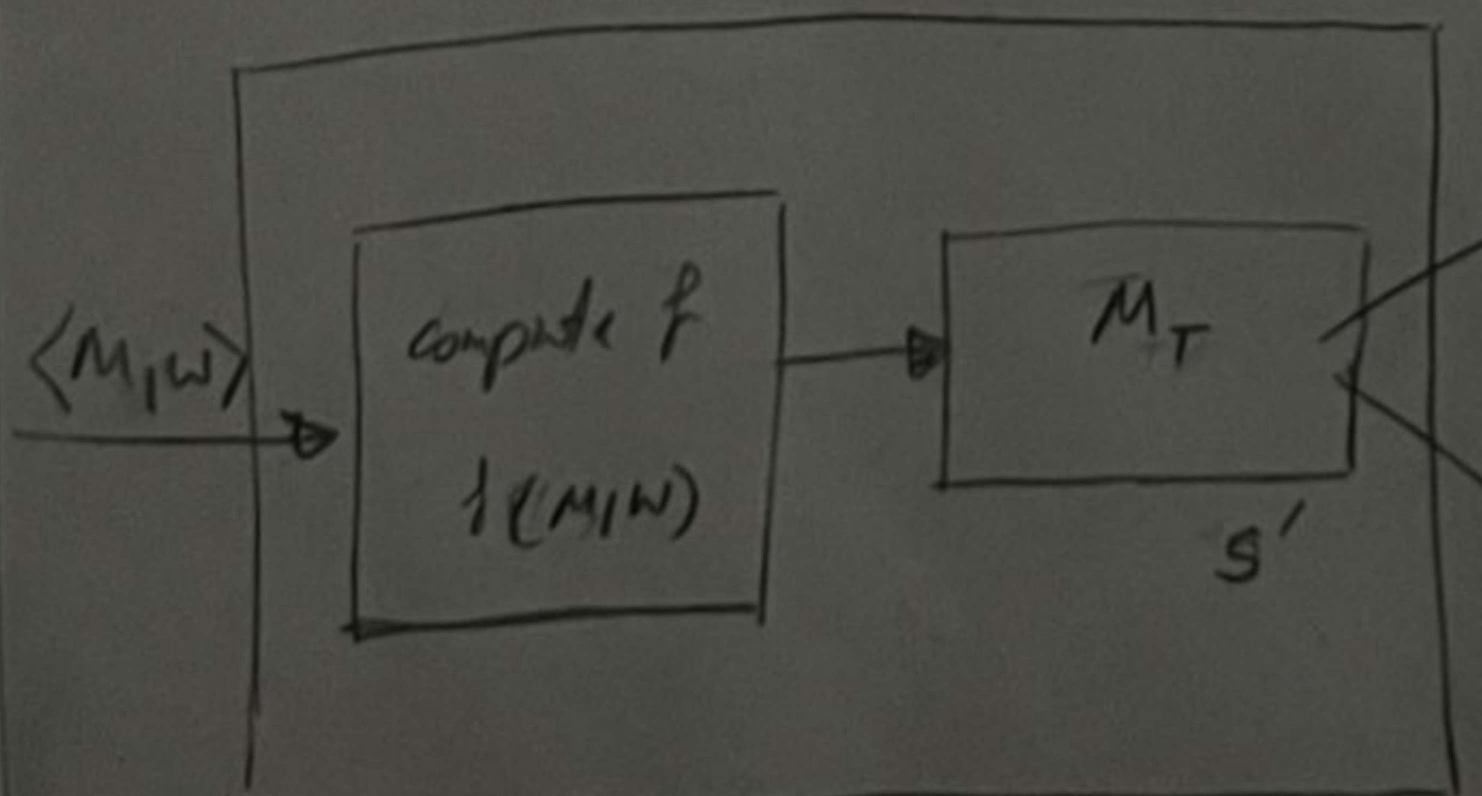
پس نتیجه می‌شود که مسئله decidable است.

سوال ۴.

$T = \{\langle M \rangle \mid M \text{ is a TM that accepts } w^r \text{ whenever it accepts } w\}$

تلاش: ثابت کنیم T undecidable است. با برهان خلف ثابت می‌کنیم. فرض خلف ما این است که T decidable است. و M_T ، T را decide می‌کند.

$$A_{TM} \leq M_T$$



اگر M ، w را accept کند، accept می‌کند

در غیر این صورت ، reject می‌کند.

S را اینطور می‌سازیم که :

TM روی input به $\langle M, W \rangle$ که w استرینگ است، و M تورینگ ماشین است،

① TM به S را می‌سازیم:

روی input به x {
- اگر x فرم ab یا ba را نداشت ← reject
- اگر x فرم ab را داشت ← accept
- اگر x، ba را داشت، M را روی w ران می‌کنیم

حال M را روی w ران می‌کنیم.

- اگر M، w را accept کند ← accept می‌کند
- در غیر این صورت ← reject می‌کند.

② M_T را روی S ران می‌کنیم و خروجی M_T را روی خروجی، نشان میدیم.

و دلیل آن که S، A_{TM} را decide می‌کند. و A_{TM} undecidable است

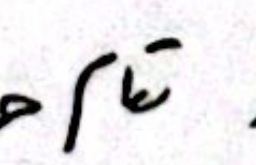
و تناقض خوردیم پس T، undecidable است.

سوال ⑤ :

$AMBIG_{CFG} = \{\langle G \rangle \mid G \text{ is an ambigous CFG}\}$

↓

2 different leftmost derivations.
(such as $S \to SS \mid \varepsilon$)

اثبات : ایده کلی این است که ما می‌خواهیم ۲ نوع derivation متفاوت داشته باشیم. که یک derivation کل string set را داشته باشد، و دیگری نیز یک کل string set را داشته باشد. و یک اشتراک‌گیری انجام میدیم بین این دو تا. 

در حقیقت ما یک عمل اشتراک‌گیری بین حروف در بالای دومینو و تمام حروف پایین دومینو انجام میدیم.

ه) ادامه:

$$P = \left\{ \begin{bmatrix} t_1 \\ b_1 \end{bmatrix}, \begin{bmatrix} t_2 \\ b_2 \end{bmatrix}, \ldots, \begin{bmatrix} t_k \\ b_k \end{bmatrix} \right\}$$

$S \to S_1 \mid S_2$

$S_1 \to \ldots$

$S_2 \to \ldots$

گرامری که S_1 ← برای ناحیه بالای دومینو است

S_2 ← " " پایین " "

فرض می‌گوییم S_1 , S_2 ، unambigous هستند.

کاری که می‌کنیم این است که S_1 و S_2 را مجبور می‌کنیم string ‌ای بسازند که برای هر دو

یکسان است اگر و تنها اگر این دومینو‌ها باهم match شوند

تنها ambiguity مخصوص پارت اول است که بین S_1 ، S_2 باید choose کنیم که کدام را

انتخاب کنیم

این گرامر (G) ، Ambig است اگر و تنها اگر:

$$t_{i_1} t_{i_2} \ldots t_{i_m} d_{i_m} \ldots d_{i_2} d_{i_1} = b_{i_1} b_{i_2} \ldots b_{i_m} d_{i_m} \ldots d_{i_1}$$

$$\Longleftrightarrow$$

$$t_{i_1} \ldots t_{i_m} = b_{i_m} \ldots b_{i_m}$$

اگر هیچ اشتراک مشترکی بین S_1 و S_2 وجود نداشته باشد، unAmbig است. ولی در غیر این صورت

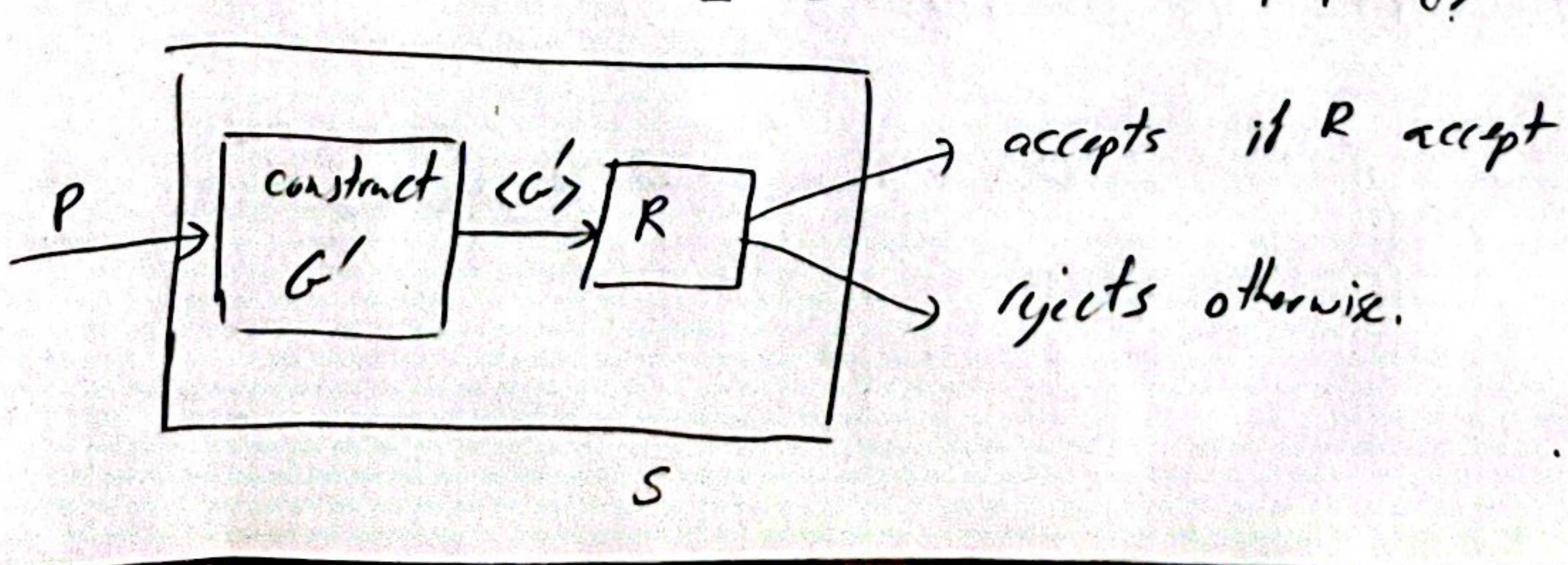
Ambig است.

درحقیقت، فرض می‌کنیم زبان $ABIG_{CFG}$ تصمیم‌پذیر است و ماشین R decider آن است. G' را به R

می‌دهیم. [اگر R ، accept کرد ← S اکسپت می‌کند

در غیر این صورت ← reject می‌کند]

پس S یک decider برای PCP است. که تناقض رسیدیم ※ ↯ $ABIG_{CFG}$ تصمیم‌ناپذیر است.



S

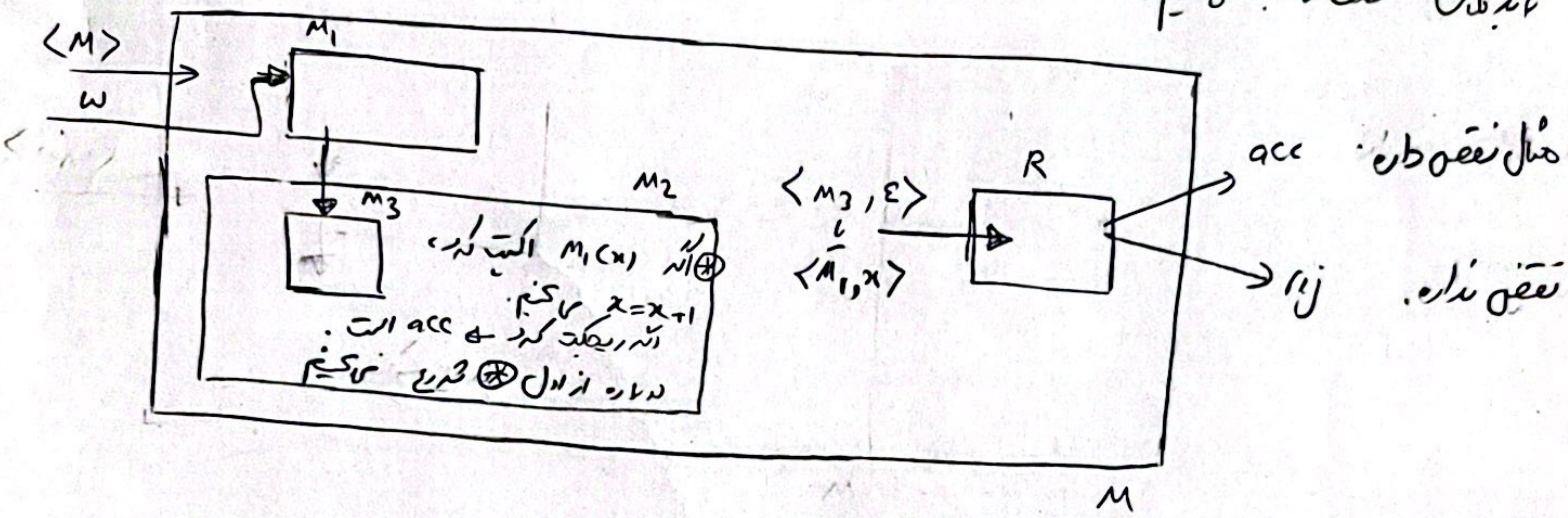
سوال ۱

$$f(x) = \begin{cases} 3x+1 & \text{for odd } x \\ x/2 & \text{for even } x \end{cases}$$

مسئله: آیا به ازای تمامی اعداد N این دنباله به یک ختم می‌شود یا نه.

ماشین R را decider برای A_{TM} در نظر می‌گیریم. که ورودی $\langle M, w\rangle$ را می‌گیرد.
(w هر عدد دلخواه است)

با برهان خلف ثابت می‌کنیم.



در این سوال M_1، input همان w را می‌گیرد. f را روی input اجرا می‌کنیم.

M_3 برای همه اعداد عضو N، عدد x را به M_1 می‌دهد، تا وقتی که accept می‌شود، عدد x را $x+1$ می‌کند و به M_1 می‌دهد تا وقتی که دیگه عدد x' جایی در M_1، acc نشود و loop بزند.

M_1

```
$ if (w == 1) → acc
  if (w is odd) w = 3w+1   goto $
  else if (w is even) w = w/2   goto $
```

در این حالت مثال نقض به دست آوردیم. از decider ATM به نام R استفاده می‌کنیم یعنی input می‌گیرد به شکل $\langle M_1, x\rangle$ را به R می‌دهد و result همان R را برمی‌گرداند.

if R reject ← M_3 accept
وگرنه loop است.

در نهایت مسئله را به یک مسئله decidable می‌خواهیم تبدیل کنیم. پس باید چک کنیم که در نهایت مثال نقض دارد یا نه. اگر دیدیم که مثال نقض پیدا شد، داریم لوپ می‌زنیم. چون می‌خواهیم M_3 decider باشد، از R استفاده می‌کنیم:
M_2 را می‌سازیم که $\langle M_1, y\rangle$ را روی R run کند. حال اگر R اکسپت کرد، یعنی مثال نقض پیدا کردیم و خواسته مسئله اثبات می‌شود و accept می‌کند (پس decide می‌کند)

سوال ۷

مسئله را به صورت زیر می‌نویسیم :

$$USELESS_{TM} = \{ \langle M, q \rangle \mid q \text{ is useless in } M \}$$

با استفاده از کاهش E_{TM} در حالت ضعف گزاره را اثبات می‌کنیم . فرض می‌کنیم TM زبان صورت سوال S باشد . حال یک decider برای E_{TM} می‌خواهیم که به شکل زیر آن را اثبات می‌کنیم :

- برای هر TM مثل M که q_{acc} باشد ⟸ بی فایده است این صفت . اگر دستخط ما ... زبان TM مثل M ، ∅ (تهی) باشد . با توجه به این نکته ، decider مثل زبان E_{TM} را می‌سازیم :

- ورودی $\langle M \rangle$ را می‌گیرد
- ماشین S ازای $\langle M, q_{acc} \rangle$ اجرا می‌کند . (q_{acc} مال M است)
- اگر S ، accept کرد ⟸ ماشین کمکی تهی را می‌پذیرد
  که در غیر این صورت reject می‌کند .

بنابراین به تناقض رسیدیم ، زیرا این decider برای E_{TM} است در صورتی که می‌دانیم چنین نیست .

⟸ پس این مسئله undecidable است .

سوال ۸ (امتیازی)

تابع busy-beaver غیر قابل محاسبه است زیرا با سرعتی بیشتر از سرعت دستگاه محاسبه ، رشد می‌کند .
اثبات ، f را یک تابع قابل محاسبه دلخواه در نظر می‌گیریم

f قابل محاسبه است ⟸ پس $F(x) = \Sigma_{0 \le i \le x} f(i) + i^2$ ، یک ماشین k استیتی است .

$$M_F : x\,1's \rightarrow F(x)\,1's$$

M را در نظر می‌گیریم : x سپس M_F سپس M_F ، زمانی که X : blank $\rightarrow$ x 1's

(می‌دانیم x شامل x استیت است)

ادامه ۸ :

M به صورت زیر رفتار می‌کند ،

- M در ابتدا x 1's را می‌نویسد.
- M ، M_F را سیمیلیت می‌کند و $F(x)$ 1's را روی tape می‌نویسد.
- M ، M_F را دوباره سیمیلیت می‌کند $F(F(x))$ 1's را روی tape می‌نویسد.

M دارای $\underline{x + 2k}$ استیت است ← $bb(n+2k) \geq$ اولین output توسط M از M

$x + F(x) + F(F(x)) = M$,

حال $F(x) \geq x^2 > x+2k$ است برای $x > m$ ، $F(x) > F(y)$ است زمانی که $x > y$ است پس: $F(F(x)) > F(x+2k) > f(x+2k)$.

پس: $bb(x+2k) \geq x + F(x) + F(F(x)) > F(F(x)) > F(x+2k) > f(x+2k)$

منبع: انترنت ۱

بنابراین f تابع دلخواه بود و bb غیر قابل محاسبه است.

به عبارت دیگر ، فرض خلف می‌کنیم که BB قابل محاسبه است ، و TM به R داریم که آن را محاسبه کند. پس R برای هر عدد n ، روی input 1^n halt می‌کند و $1^{BB(n)}$ را روی tape دارد. حال TM به S را تعریف می‌کنیم: S با شروع از یک blank براساس R ، ... halt می‌کند.

در مرحله اول ، روی tape n تا 1 می‌نویسد. پس S تعداد 1 های موجود را ۲ برابر می‌کند.

در مرحله سوم ، S ، R را روی input 1^{2n} ران می‌کند. پس $S = 1^{BB(2n)}$

اگر با blank-tape شروع شود halt می‌کند. برای اجرای S ، n استیت نیاز داریم ولی ادامه کار ، 9 استیت می‌خواهیم.

$BB(n+9)$ ← بیشترین 1 هایی که TM با $(n+9)$ استیت halt می‌کند

$BB(n+9) \geq BB(2n)$. می‌دانیم $BB(k)$ تابع صعودی است پس بنابراین

$\underline{BB(n+9) < BB(2n)}$ است پس به تناقض رسیدیم و ۱۰۰٪ $BB(k)$ غیرقابل محاسبه باشد.

زیرا $n > 9$ است.