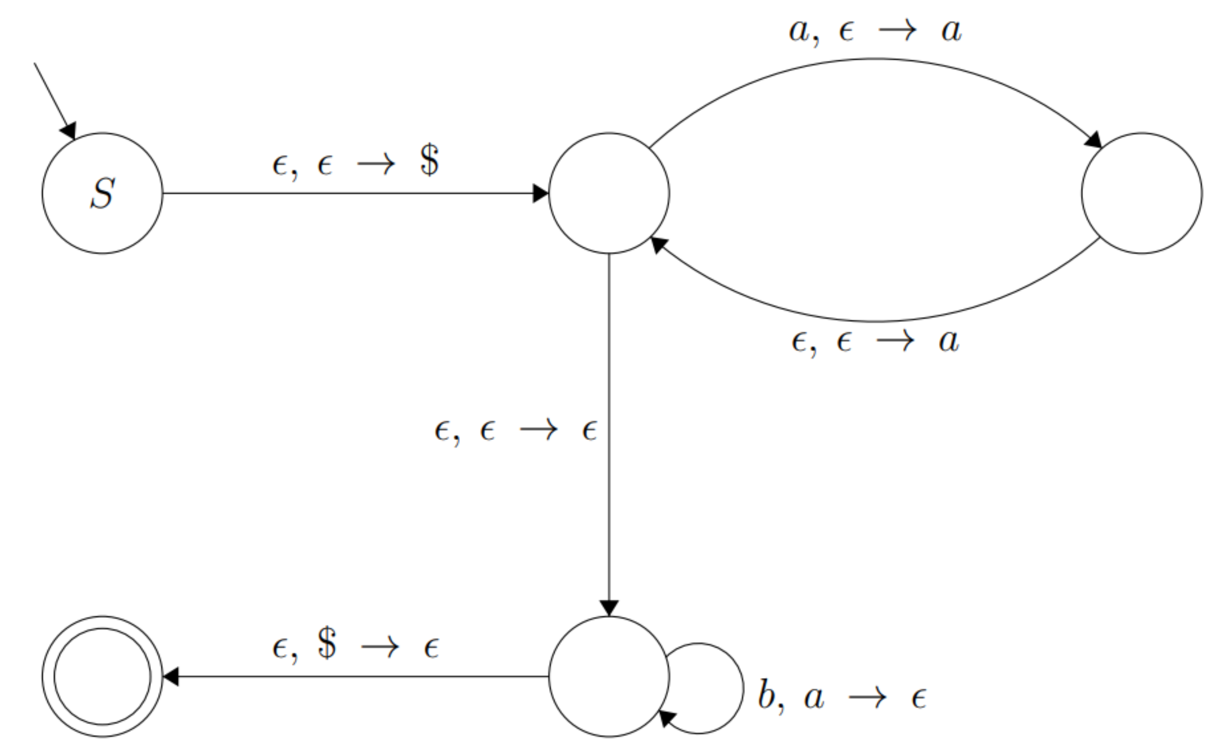
بسمه تعالی

نظریه زبان‌ها و ماشین‌ها

پاسخ تکلیف چهارم

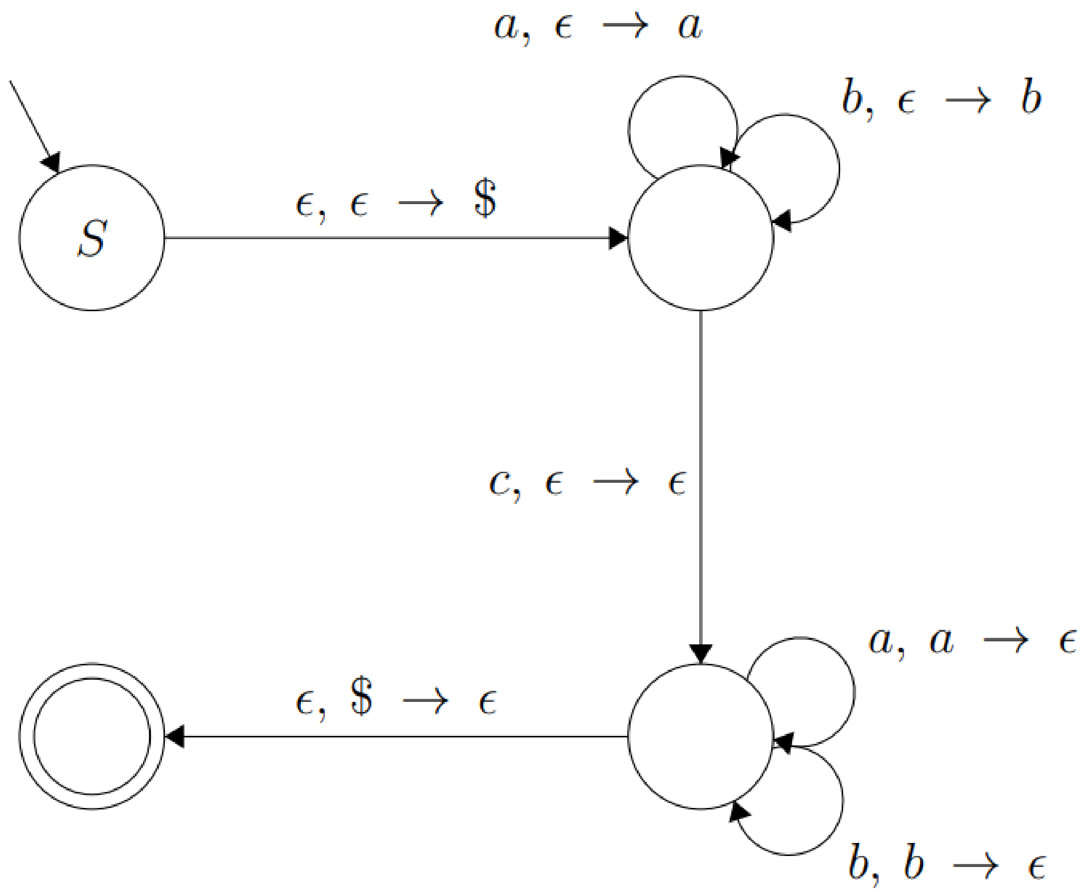
۱. برای زبان های زیر PDA بر روی الفبای $\sum = \{a, b, c\}$ طراحی کنید

$L1 = \{a^n b^{2n} : n \geq 0\}$ $\sum = \{a, b\}$



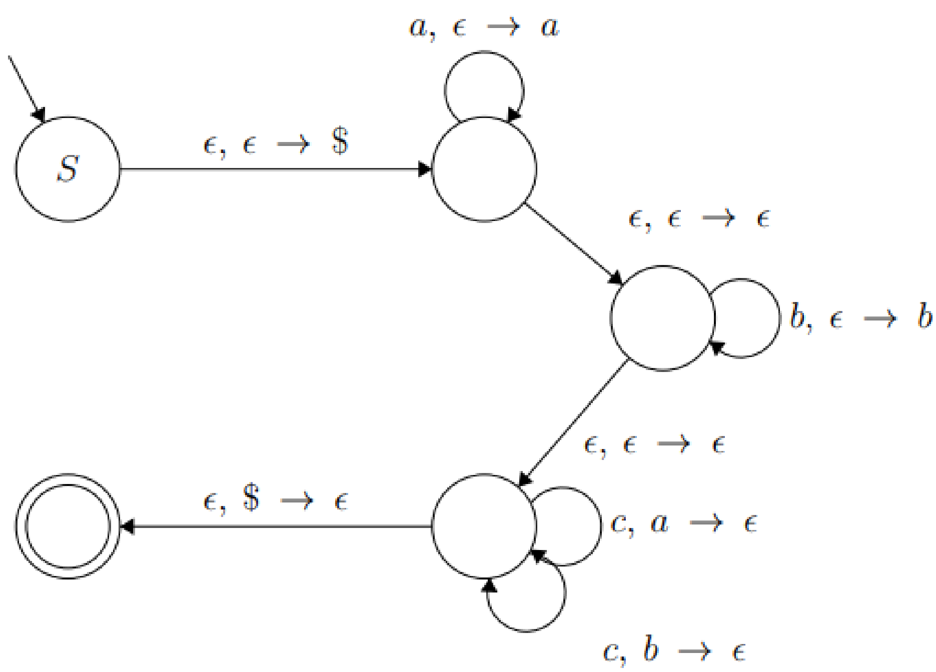
در این شکل، در اول کار به پشته نماد $\$$را اضافه می کنیم تا انتهای آن را مشخص کنیم. سپس، به ازای هر aای که می خوانیم، دو نماد aبه پشته اضافه می کنیم. توجه کنید که به محض این که یکb دیده شود از یال اپسیلون عمودی باید عبور کنیم. در آن جا، به ازای هر bای که دیده می شود، یکa از پشته حذف می کنیم. در اینجا اگر در ورودی aدیده شود، Rejectمی شود. اما اگر پشته خالی شده باشد و نماد $\$$را ببینیم، به حالت پذیرش می رویم. در اینجا هیچ حرفی در ورودی نباید باقی مانده باشد، چون که در این صورت Rejectمی شود. در غیر این صورت، Acceptمی شود. توجه کنید که این موضوع که ما برای هر دو نماد a به پشته اضافه کردیم و به ازای هر aدر پشته یک bمی خوانیم، معادل شرط این است که تعداد bها دو برابر تعداد aهای داخل رشته ی ورودی باشد.

$L2 = \{wcw^R : w \in \{a, b\}^*\}$



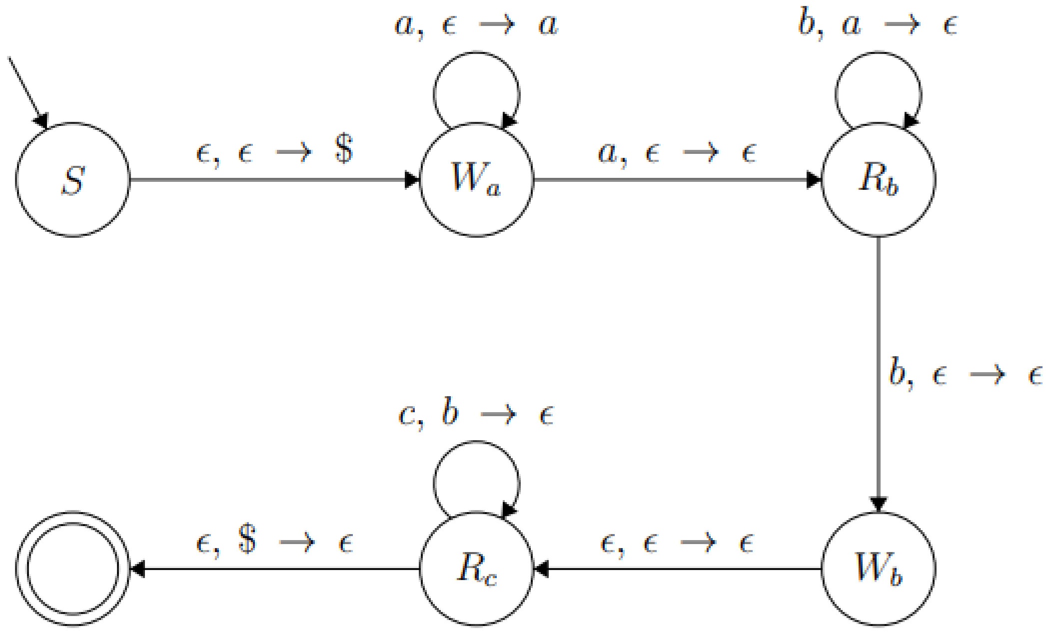
در این شکل، در اول کار به پشته نماد $را اضافه می کنیم تا انتهای آن را مشخص کنیم. سپس، به ازای هر حرفی که خواندیم، آن حرف را به پشته اضافه می کنیم. به محض این که یک cدیده شود، از یال عمودی به بخش پایینی PDAخواهیم رفت. در بخش پایینی، به ازای حرفی که در اول پشته است، همان حرف را در ورودی می خوانیم. به دلیل خاصیت Frist in First Outمربوط به پشته، این عمل معادل این است که چک کنیم رشته ی سمت راست cبرابر برعکس رشته ی سمت چپ cباشد. در اینجا اگر حرف cببینیم، یا رشته ای که می بینیم طبق برعکس رشته ی سمت چپ cنباشد، Rejectمی کنیم. چون جایی برای رفتن نخواهیم داشت. در آخر اگر پشته خالی شده باشد و نماد $را ببینیم، به حالت پذیرش می رویم. در اینجا هیچ حرفی در ورودی نباید باقی مانده باشد، چون که در این صورتReject می شود. در غیر این صورت، Acceptمی شود. همانطور که مشاهده می شود، این همان رفتاری که از زبان Lانتظار داریم را از خود نشان می دهد.

$L3 = \{a^n b^m c^{n+m} : n \geq 0, m \geq 0\}$



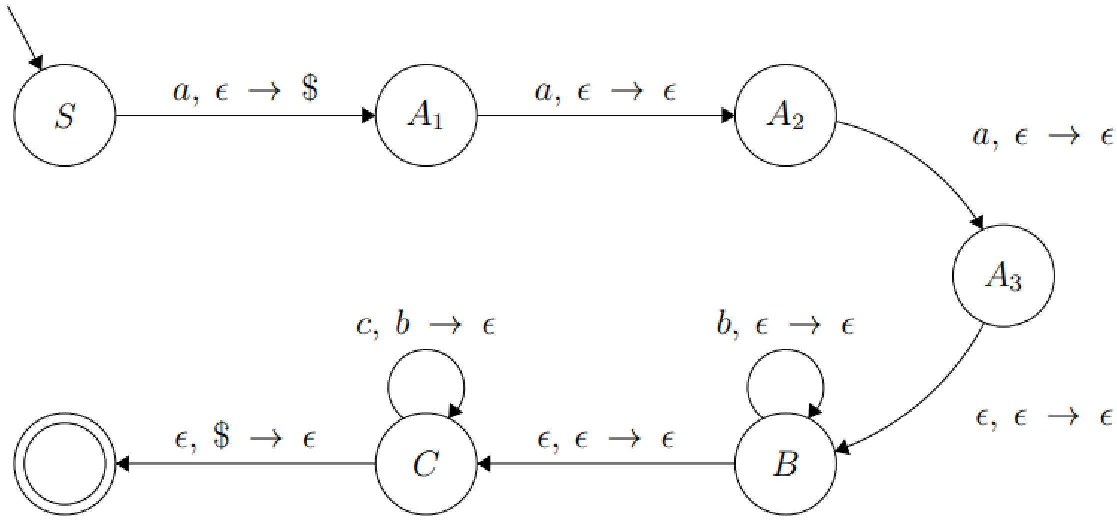
در این شکل، در اول کار به پشته نماد $\$$را اضافه می کنیم تا انتهای آن را مشخص کنیم. سپس، به ازای هر aای که می خوانیم و تا زمانی که حرف bنخوانده باشیم، نماد aرا به پشته اضافه می کنیم .توجه کنید که اینجا اگر حرفً cدر ورودی ببینیم، حتما باید به رأس پایینی برویم. اگر هم حرف b ببینیم حتما باید به رأس میانی برویم. در رأس میانی هم به ازای هر bای که می خوانیم، یک حرف bبه پشته اضافه می کنیم. توجه کنید که در اینجا اگر حرفً aببینیم، ورودی حتما Rejectمی شود. اگر هم حرف c ببینیم، حتما به رأس پایینی باید برویم. در رأس پایینی هم تا زمانی که بالای پشته نماد هایa یا bباشد، از ورودی حرف cرا می خوانیم. توجه کنید که در اینجا اگر در ورودی حرف aیا bدیده شود Rejectمی کنیم. این رفتار ها در مجموع به این معنی خواهند بود که رشته ی ورودی باید حتما تعدادی aپشت سر هم و تعدادی bپشت سر هم باشد. در آخر هم تعدادی cپشت سر هم داشته باشیم. تعداد aهای خوانده شده و تعداد آن ها که در پشته هستند برابر هستند. همین خاصیت هم برای تعداد bها برابر است. به همین دلیل، تعدادً cهایی که می خوانیم حتما برابر با مجموع تعداد aها و bها است. در آخر اگر پشته خالی شده باشد و نماد $\$$را ببینیم، به حالت پذیرش می رویم. در اینجا هیچ حرفی در ورودی نباید باقی مانده باشد، چون که در این صورت Rejectمی شود. در غیر این صورت،Accept می شود. همانطور که مشاهده می شود، این همان رفتاری که از زبان Lانتظار داریم را از خود نشان می دهد.

$L4 = \{a^n b^{n+m} c^m : n \geq 0, m \geq 1\}$



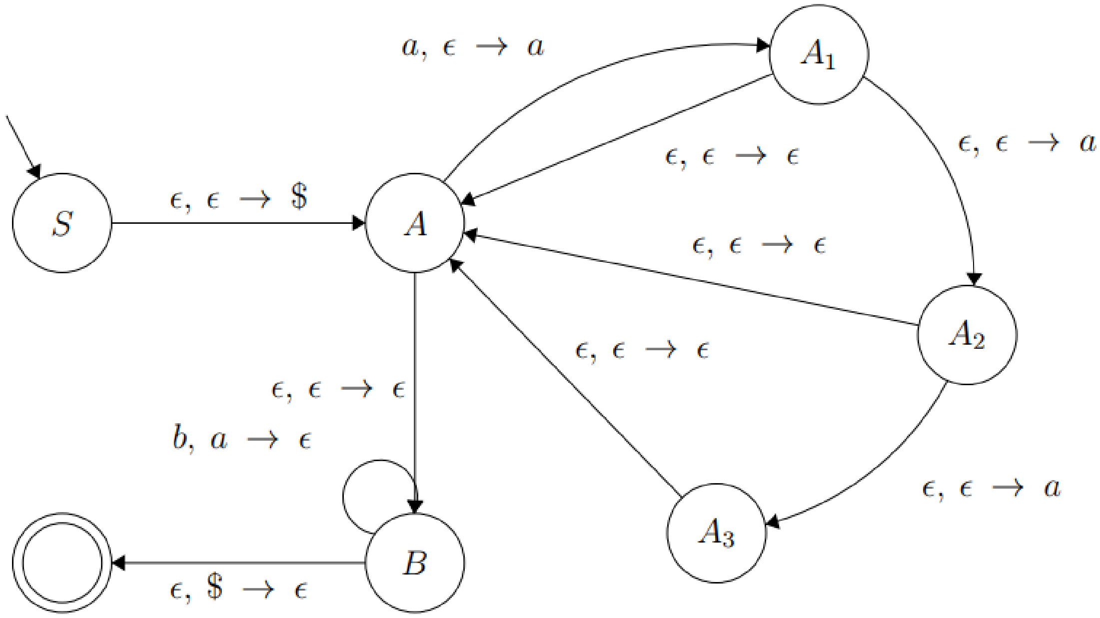
در این شکل، در اول کار به پشته نماد $\$$ را اضافه می کنیم تا انتهای آن را مشخص کنیم. سپس به ازای هر a ای که می خوانیم، یک a به پشته اضافه می کنیم. توجه کنید که در رأس Wa اگر حرف b دیده شود مجبور می شویم که به Rb برویم. اگر هم حرف c دیده شود Reject می کنیم. به محض این که یک b دیده شود و ما در Wa باشیم، به رأس Rb می رویم. تا زمانی که در پشته حرف a بود، حرف اول پشته را حذف می کنیم و حرف b را از ورودی می خوانیم. در زمانی که a های داخل پشته تمام شدند، به رأس Wb می رویم و به ازای هر b ای می خوانیم یک حرف b به پشته اضافه می کنیم. در اینجا، هر موقع حرف c ببینیم به رأس Rc می رویم و به ازای هر b داخل پشته، یک c از ورودی می خوانیم .به دلیل یال عمودی، در این مرحله حداقل یک b باید خوانده شود و شرط $m > 0$ را برقرار می کند .توجه کنید که به دلیل شکل، PDA هر رشته ای که به فرم $a^*b^+c^+$ نباشد Reject میشود. در آخراگر پشته خالی شده باشد و نماد $\$$ را ببینیم، به حالت پذیرش می رویم. در اینجا هیچ حرفی در ورودی نباید باقی مانده باشد، چون که در این صورت Reject می شود. در غیر این صورت، Accept می شود. چون که ما به ازای هر a یک b را می خوانیم و به ازای هر b اضافه بر آن ای، یک c می خوانیم، می فهمیم که تعداد b ها برابر با مجموع a و c است. پس میفهمیم که این PDA همان رفتاری که از زبان L انتظار داریم را از خود نشان می دهد.

$L5 = \{a^3b^nc^n : n \geq 0\}$



در این شکل، در اول کار به پشته نماد $را اضافه می کنیم تا انتهای آن را مشخص کنیم. در اینجا یک حرف aهم از ورودی می خوانیم. به همین صورت و با فرمت DFA، دو حرف دیگر aاز ورودی می خوانیم. توجه کنید که در این سه حرف اول، اگر هر حرفی به جز aدر ورودی باشد Rejectمی کنیم. بعد از خواندن سه حرف a، از رأس A3خارج می شویم و یک تعداد دلخواه bاز ورودی می خوانیم. به ازای هر حرف b، یک حرف bرا به پشته اضافه می کنیم. به محض این که یک حرف cدر ورودی دیده شود و ما در رأس Bبودیم، به رأس Cباید برویم و به ازای هر حرف cیک حرفb از پشته را پاک می کنیم. این منطق برابری تعداد bها و cرا تضمین می کند. در آخر اگر پشته خالیشده باشد و نماد $را ببینیم، به حالت پذیرش می رویم. در اینجا هیچ حرفی در ورودی نباید باقی مانده باشد، چون که در این صورت Rejectمی شود. در غیر این صورت، Acceptمی شود. شکل این PDAبه ما نشان می دهد که هر رشته ای که به فرم a3b*c*نباشد Rejectمی شود. همانطور که مشاهده می شود، این همان رفتاری که از زبان Lانتظار داریم را از خود نشان می دهد.

$L6 = \{a^n b^m : m \geq n + 2\}$ $\sum = \{a, b\}$



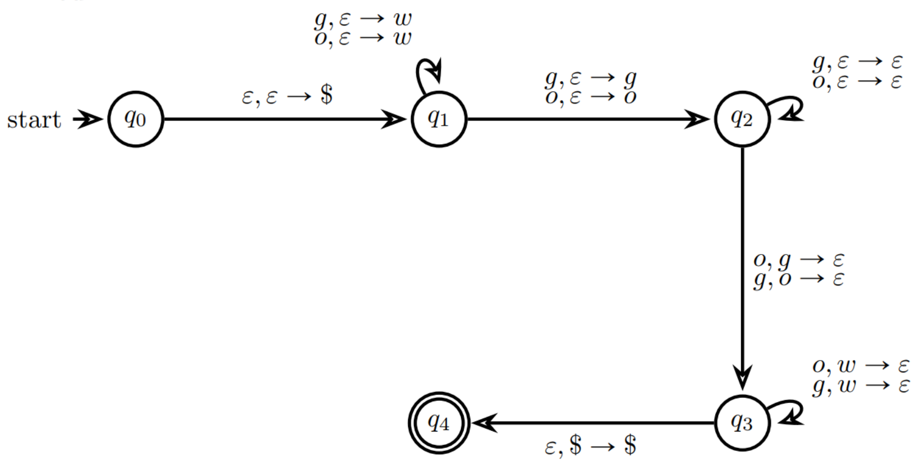
در این شکل، در اول کار به پشته نماد $ را اضافه می کنیم تا انتهای آن را مشخص کنیم. سپس به رأس A می رویم. در این رأس، هر حرف a ای که دیدیم به رأس A1 می رویم. در اینجا، حداقل یک حرف a از ورودی خوانده ایم و یک a به پشته اضافه کرده ایم. در ادامه، با رفتن به رأس A2، یک حرف a دیگر به پشته اضافه می کنیم و با رفتن به A3 هم یک حرف a دیگر به پشته اضافه می کنیم .درواقع با استفاده از رئوس Ai، برای هر a در ورودی حداقل یک a و حداکثر سه a به پشته اضافه می کنیم. درواقع در نهایت با خواندن n عدد a، یک تعدادی بین n و 3n حرف a در پشته خواهند بود. به محض این که یک حرف b ببینیم، باید به رأس B برویم. در این رأس هر b ای که می بینیم را با یکی از حروف a داخل پشته Match می کنیم. به این صورت، تعداد b هایی که می خوانیم برابر با تعداد a هایی خواهد بود که در پشته هست. همچنین توجه کنید که اگر حرف a ای ببینیم ورودی Reject می شود. در آخر اگر پشته خالی شده باشد و a در آن نداشته باشیم، نماد $ را ببینیم، به حالت پذیرش می رویم. در اینجا هیچ حرفی در ورودی نباید باقی مانده باشد، چون که در این صورت Reject می شود. در غیر این صورت، Accept می شود. همانطور که مشاهده می شود، این همان رفتاری که از زبان L انتظار داریم را از خود نشان می دهد. چون تعداد b هایی که خوانده ایم را برابر با آن تعدادی که بین n و 3n بود قرار دادیم.

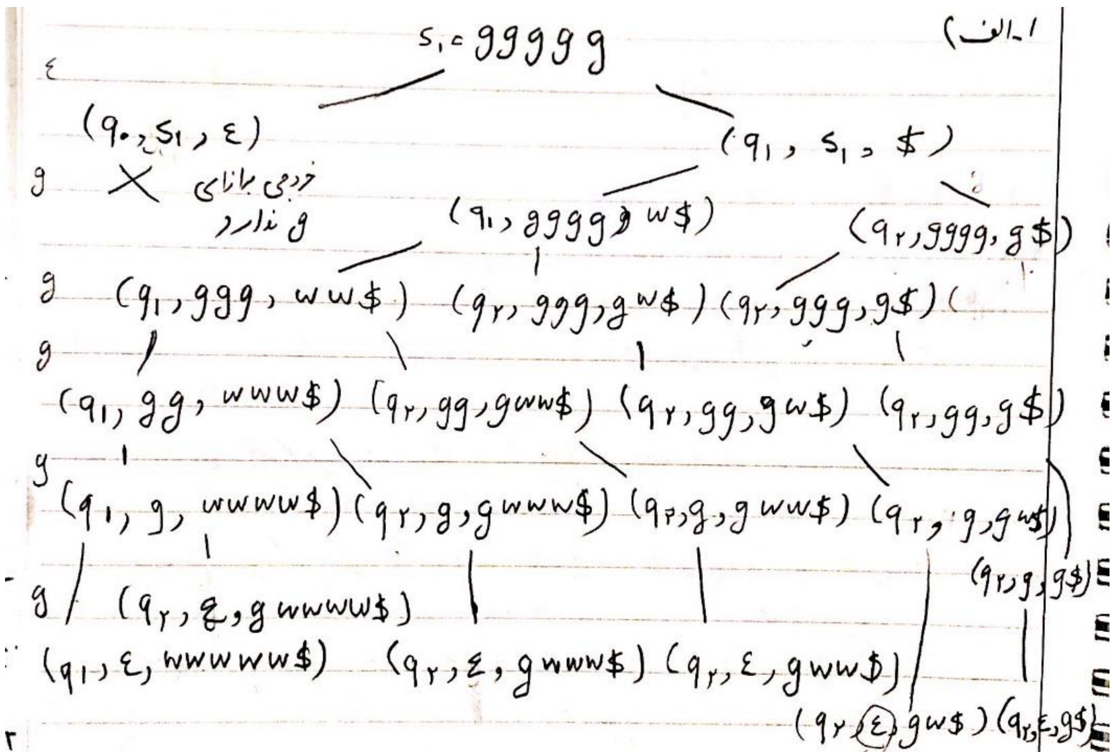
بسمه تعالی

۲. یک M PDAو دو رشتهٔ s1 = ggggg و s2 = oogoogggggg در اختیار شما قرار گرفته است.

الف. درخت محاسبهٔ نظیر رشتهٔ s1 را به طور کامل ترسیم کنید.

Your PDA M:





همین طور که می‌بینید رشته پایان یافت اما به یک استیت فاینال نرسیدیم

پس این رشته توسط این ماشین پذیرفته نمی‌شود

بسمه تعالی

ب. دربارهٔ پذیرش یا عدم پذیرش رشتهٔ s2توسط M PDA، با بیان استدلال، اظهار نظر کنید. اگر $s2 \in L(M)$ آنگاه کانفگوریشن هایی که طی می شوند تا پذیرش صورت گیرد را معین کنید. (ترسیم درخت محاسبه در قسمت ب لازم نیست.)

$(q_0, s_2, \varepsilon) \vdash (q_1, s_2, \$) \vdash (q_2, gg00ggggg, 0\$) \vdash$

$(q_2, g00ggggg, 0\$) \vdash (q_2, 00ggggg, 0\$) \vdash$

$(q_2, 0ggggg, 0\$) \vdash (q_2, ggggg, 0\$) \vdash (q_2, gggg, 0\$)$

$\vdash (q_2, ggg, 0\$) \vdash (q_2, gg, 0\$) \vdash (q_2, g, 0\$) \vdash$

$(q_3, \varepsilon, \$) \vdash (q_4, \varepsilon, \$)$

به حالت فاینال رسیدیم پس پذیرش می‌شود

اما چرا؟ در واقع این پی دی ای به این صورت عمل می‌کند که حتما باید
آرک های غیر لوپ را غیر طوقه ها را ببینیم $q_0 \to q_1$ که استک خالی را و $q_3 \to q_4$ نیز
همین را نمایان می‌کند پس جز آرک $q_1 \to q_2$ می‌گوید هر حرفی خواندی همان را در استک
بگذار و $q_2 \to q_3$ می‌گوید در این انتها باید هر چیزی خواندی مثل آن را از استک بردار
پس ما حتما دو حرف متمایز در این رشته می‌خواهیم چون مثلاً اگر g خوانده و در استک گذاشته باشیم
بعداً برای خالی کردن استک باید 0 بخوانیم رشته: بین این دو حرف متمایز هم مهم نیست چون
آرک $q_2 \to q_2$ استک را تغییر نمی‌دهد اما نکته‌ای که هست قبل و بعد این دو حرف متمایز
باید به یک اندازه باشد زیرا آرک $q_1 \to q_1$ هر چیزی را بخواند ها را استک می‌گذارد
و آرک $q_3 \to q_3$ هر چه را بخواند ها از استک برمی‌دارد و چون در آخر استک
باید خالی باشد این تعداد ها باید یکی باشد پس قبل و بعد دو حرف متمایز
باید هم سایز باشند.

به طور خلاصه این PDA همه رشته‌هایی که دو حرف متمایز دارند که یکی از اول
از اول رشته فاصله مشخص دارد و دیگری از آخر رشته همان فاصله را دارد می‌پذیرد

بسمه تعالی

۳. ثابت کنید زبان‌های زیر مستقل از متن نیستند. .(یک مورد اختیاری)

$L1: \{a^n b^m : n \leq m^2\}$

برای اثبات، فرض کنید Lمستقل از متن باشد. در نتیجه، می فهمیم که لم پامپینگ برای آن برقرار است. پس یک pبرای لم پامپینگ داریم. حال رشته ی به فرم $a^{(2p)*(2p)}b^{2p}$ را در نظر بگیرید و آن را به فرم vwxyzبشکنید. حال، روی جایی از رشته که wxyقرار می گیرد حالت بندی می کنیم:

⁕) w و yهردو در بخش مربوط به aها هستند: در این حالت، می فهمیم که اگر رشته ی vwwxyyzرا در نظر بگیریم، فقط تعداد aهای آن بیشتر از قبل خواهد شد. پس تعداد aهای آن که قبلا برابر با m۲بود، الآن بیشتر شده است. پس رشته در زبان نیست که با فرض مستقل از متن بودن در تناقض است.

⁕) w و yهر دو در بخش مربوط به bها هستند: در این حالت می فهمیم که اگر رشته ی vxzرا در نظر بگیریم، فقط تعداد bهای آن از رشته ی اول کمتر است و تعداد aهای آن ثابت مانده است. این به این معنی است که تعداد aهای این رشته همان m^2است، اما تعداد bها کمتر از mشده است. پس دیگر شرط عضویت در Lبرقرار نیست که با فرض مستقل از متن بودن در تناقض است.

⁕) w و yهم در bها و هم در aها تأثیر دارند: در این حالت، فرض کنید که تعداد bها و aها دررشته های wو yمجموعا Bو Aباشد. می دانیم که $A \leq |wxy| \leq p$و طبق حالت فعلی، $A, B \geq 1$.پس اگر رشته ی vxzرا در نظر بگیریم، می فهمیم که تعداد bهای آن Bتا و تعداد aهای آن Aتا کم شده است. پس داریم:

$n_a(vxz) = (2p)^2 - A \qquad n_b(vxz) = 2p - B$

طبق نابرابر های مذکور، داریم:

$n_a(vxz) \geq (2p)^2 - p \qquad n_b(vxz) \leq 2p - 1$

$\rightarrow (2p - 1)^2 = 4p^2 - 4p + 1 \geq n_b(vxz)^2$

$\rightarrow (2p)^2 - p = 4p^2 - p \leq n_a(vxz)^2$

چون می دانیم که کران پایین $n_a(vxz)$ از کران بالای $n_b(vxz)^2$ بزرگ تر است، می فهمیم که $n_a(vxz) > n_b(vxz)^2$.پس می فهمیم که ، vxz در زبان نیست با فرض مستقل از متن بودن در تناقض است.

در نتیجه در کل به تناقض می خوریم و می فهمیم که Lمستقل از متن نیست.

بسمه تعالی

$L2: \{a^{n!} : n \geq 0\}$

رشته $a=a^{m!}$ را در نظر بگیرید که m اولین عدد بزرگتر از طول تزریق باشد فرض خلف کنیم که زبان مستقل از متن است هرجوری که تقسیم بندی کرده باشیم $v=a^k$ و $y=a^l$ میباشد پس $w0=uxz$ طولی برابر $m!-k-m$ دارد که تنها در صورتی در زبان است که $w0=j!$ شود اما این ممکن نیست چون $k+l<=m$ است و $m-k-l>(m-1)!$

پس به تناقض رسیدیم و زبان مستقل از متن نیست.

$L3: \{a^n b^m : n \text{ is prime or } m \text{ is prime}\}$ حل این سوال اختیاری می باشد.

این زبان مستقل از متن نیست. برای اثبات این، فرض کنید Lمستقل از متن باشد. همچنین فرض کنیدGیک گرامر فرم نرمال چامسکی برای Lباشد که |V| متغیر دارد. رشته ی ab^qرا در نظر بگیریدکه qاولین عدد اول بزرگ تر از $2^{|V|+3}$است. درخت اشتقاق آن را Tبنامید. اگر متغیر برگی که پایانه ی aرا ایجاد می کند حذف کنیم، می فهمیم که یک درخت داریم که تعداد بچه های هر رأس حداکثر ۲است و حداقل $2^{|V|+3} - 1$رأس دارد. پس می فهمیم که در این درخت کوچک تر، یک مسیر داریم که از ریشه شروع می شود و حداقل $|V| + 2$رأس در آن دارد. چون ما در کل |V|متغیر داشتیم، می فهمیم حداقل یک متغیر vداریم که در این مسیر دو بار تکرار شده است و یکی جد دیگری است. پس به شکل مشابه با شکل ۱می رسیم. اگر مثل ایده ی اثبات لم پامپینگ آن را پامپ کنیم به شکل هایی شبیه شکل ۲می رسیم. اما چون این درخت، رئوسی را داست که حداقل یک bتولید میکردند، می فهمیم رشته هایی که پامپ می شوند حداقل یک bرا دارند. اما ممکن است که aداشته باشند یا نداشته باشند. فرض کنیم این رشته هایی که با یک عمل پامپ کردن اضافه می شوند،Bعدد bو Aعدد aدارند. این دو حالت را به صورت زیر بررسی می کنیم: (توجه کنید که همین دو حالت را داریم، چون کلا یک عدد aدر رشته ی خود داشتیم)

$A = 0$:در این حالت، با هر مرحله پامپ کردن، به رشته ی ما Bعدد bاضافه می شود. اگر این رشته را qبار پامپ کنیم، تعداد bها در نهایت برابر خواهد بود با $Bq + q = (B + 1)q$که عددی اول نیست. همچنین، تعداد aها ثابت باقی مانده است و برابر با یک است. پس می فهمیم که n_aو n_bهیچکدام اول نیستند و به یک رشته می رسیم که باید در Lباشد، اما شرایطش را ندارد. پس به تناقض می رسیم.

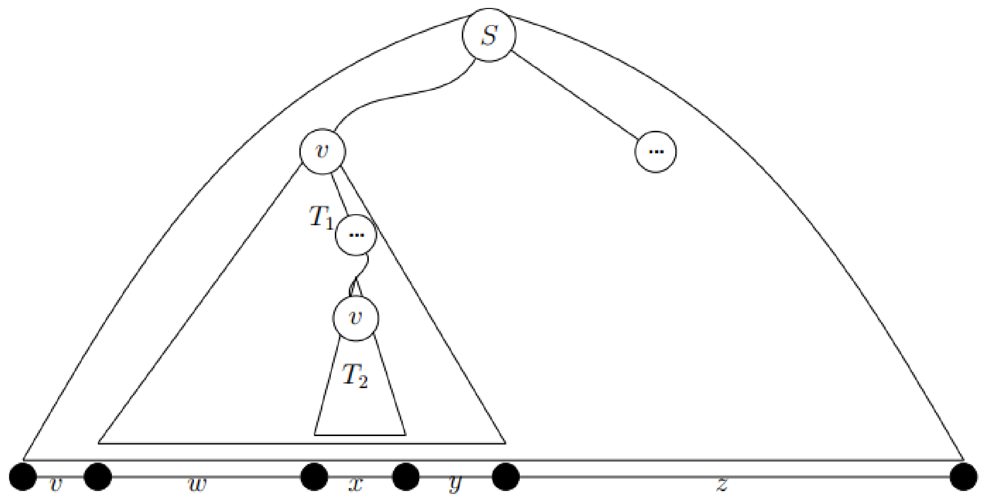
$A = 1$:در این حالت، با هر مرحله پامپ کردن، به رشته ی ما Bعدد bو یک عدد aاضافه می شود. اگر این رشته را qبار پامپ کنیم، خواهیم داشت $n_a = 1 + q$ $n_b = Bq + q = (B + 1)q$

همانطور که مشاهده می کنیم، n_bعددی اول نیست و n_aهم برابر با یک عدد اول به علاوه ی ۱است، که می دانیم عددی زوج خواهد بود. پس می فهمیم که n_aو n_bهیچکدام اول نیستند و به یک رشته می رسیم که باید در Lباشد، اما شرایطش را ندارد.

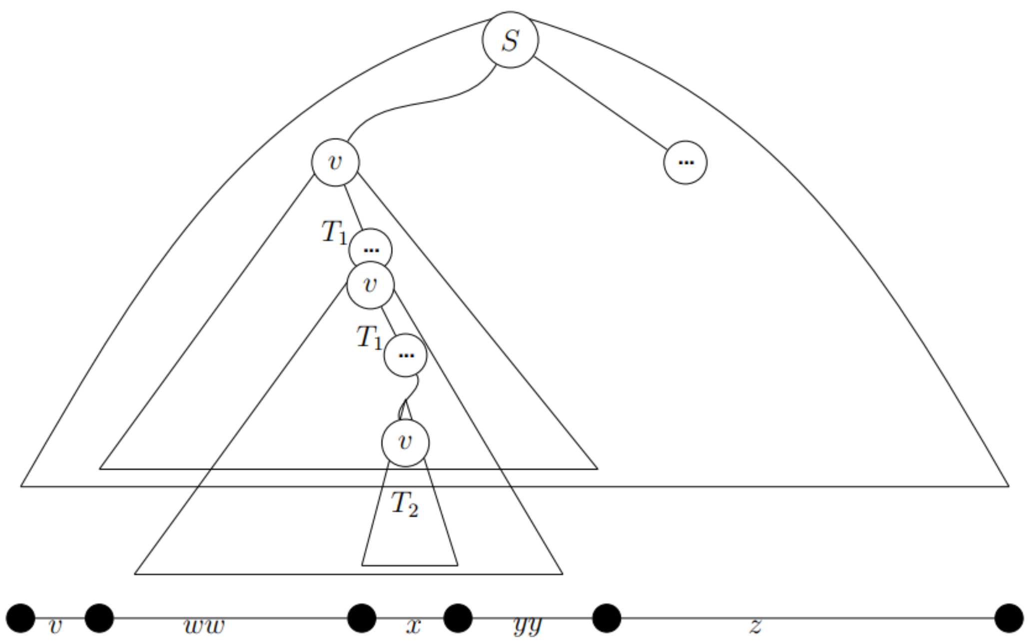
بسمه تعالی

پس به تناقض می رسیم.

در نتیجه در کل به تناقض می خوریم و می فهمیم که Lمستقل از متن نیست



شکل ۱: درخت اشتقاق مربوط به گرامر G_C. رشته‌هایی که در نهایت توسط هر زیر درخت به دست می‌آیند را هم مشخص کرده‌ایم.

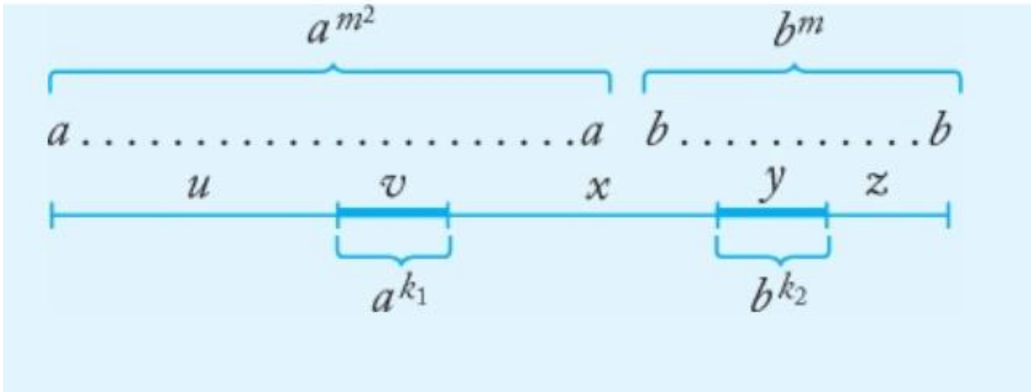


شکل ۲: درخت اشتقاق بزرگ شده با یک بار replace کردن.

بسمه تعالی

L4: $\{a^n b^j : n = j^2\}$

رشته $a^{m*m}b^m$ را درنظر بگیرید.برای رسیدن به تناقض چند راه داریم.راهی که فکر بیشتری نیاز دارد در شکل زیر نشان داده شده است:



i بار پامپ کردن به یک رشته ی جدید با $m^2+(i-1)k_1$ تعداد a و $m+(i-1)k_1$ تعداد b ختم خواهد شد. اگر برای تناقض $k_1 \neq 0$ و $k_2 \neq 0$ را در نظر بگیریم، می‌توانیم i را 0 انتخاب کنیم به همین جهت:

$(m+k_2)^2 \leq (m+1)^2 = m^2-2m+1 < m^2-k_1$

جواب در L نیست.حتی اگر $k_1 = 0$ و $k_2 \neq 0$ یا $k_1 \neq 0$ و $k_2 = 0$ در نظر می‌گرفتیم باز هم با انتخاب i=0 رشته ی پامپ شده باز هم در L نخواهد بود. پس نتیجه می‌گیریم که L یک زبان مستقل از متن نیست.

L5: $\{w \mid n_a(w) < n_b(w) \cdot n_c(w)\}$

زبان $L = \{ w \mid n_a(w) < n_c(w) \text{ and } n_a(w) < n_b(w) \}$ زیر مجموعه نامتناهی از L_1 است. درنتیجه با اثبات مستقل از متن نبودن L می توان ثابت کرد که L_1 نیز مستقل از متن نیست. حال اگر $s = a^p b^{p+1} c^{p+1}$ در نظر بگیریم، با توجه به لم تزریق و شروط آن vy می تواند شامل aوb و یا شامل bوc باشد. در حالت اول اگر uv^2xy^2z را در نظر بگیریم در کمترین حالت تعداد a ها با c ها مساوی است و یا تعداد a ها بیشتر است. در حالت دوم اگر uv^0xy^0z را در نظر بگیریم در بیشترین حالت تعداد b ها و یا c ها با تعداد a ها برابر می شود و یا کمتر. درنتیجه L و L_1 مستقل از متن نیستند.

۴. برای زبان های زیر ماشین تورینگ طراحی کنید. (توصیف سطح بالا کفایت میکند.) $\Sigma = \{a, b\}$

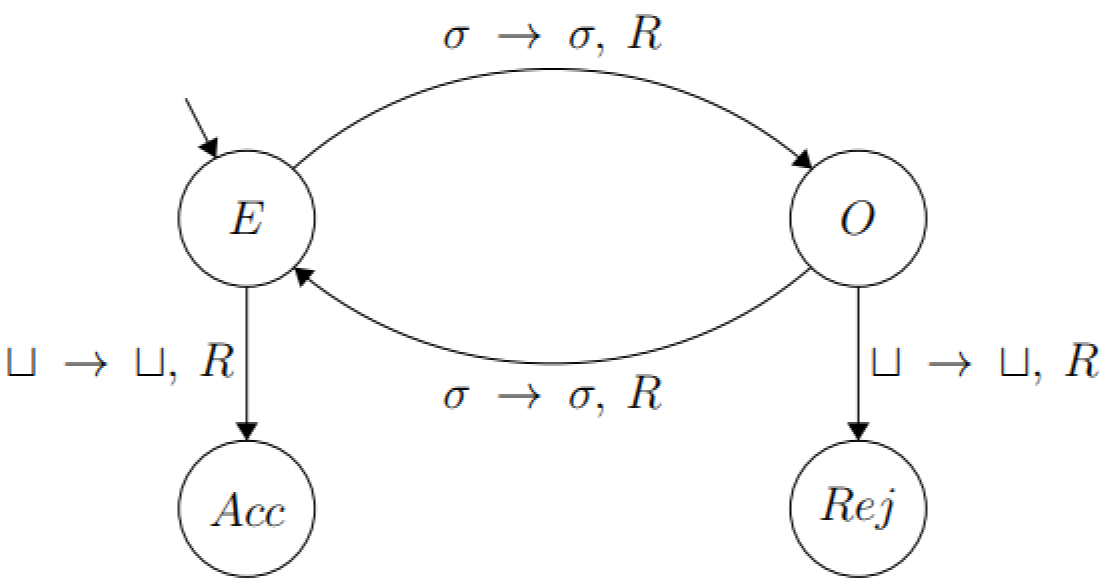
L1 = {w : |w| is even}

برای چک کردن این موضوع، از ماشین تورینگ مثل یک DFAاستفاده می کنیم. این اتومات را در شکل می توانید مشاهده کنید.

در شکل منظور از حرف σاین است که یال مذکور را برای هر σدرΣ کپی کنیم و در نظر بگیریم. منطق این ماشین تورینگ نیز به این صورت است که تا زمانی که در نوار در حال خواندن ورودی است و به خانه ی خالی نرسیده است، ورودی را بخواند و Headرا به سمت راست ببرد. در اول کار، در حالت Eهستیم که مربوط به $|w| = 2k$ است. حالت Oنیز مربوط به $|w| = 2k + 1$است. با

خواندن هر حرف از حالت Eبه Oو برعکس می رویم تا زوجیت ورودی را بتوانیم تشخیص دهیم. در نهایت، اگر در Eاگر به انتهای ورودی رسیدیم یعنی ورودی طولی زوج داشته است و آن را می پذیریم. اگر هم این اتفاق در Oافتاد یعنی طول فرد داشتیم و باید آن را Rejectکنیم.پس این ماشین تورینگ همان رفتاری که انتظار داریم را از خود به نمایش می گذارد و زبان آن همان Lاست.

پس به خواسته ی مسئله رسیده ایم.



$L2 = \{a^n b^m : n \geq 1, n \neq m\}$

توصیف سطح بالای این ماشین تورینگ به صورت زیر است:

۱حرف اول نوار را با یک نقطه Markمی کنیم.

۲با استفاده از دو حالت و با منطق DFAمانند، چک می کنیم که ورودی به حالت +a+b باشد.

اگر نبود، Rejectمی کنیم.

۳سپس، به اول نوار بر می گردیم و نقطه ی آن را از حرف اول حذف می کنیم.

۴به ازای هر aبه صورت زیگ زاگی آن را با نقطه Markمی کنیم و یک bاز سمت راست را هم به همین شکل Markمی کنیم.

اگر bای پیدا نکردیم و به خانه ی خالی رسیدیم، Accept می کنیم.

۵به چپ ترین aای که نقطه ندارد باز می گردیم. (این موضوع را با دیدن اولین aبا نقطه تشخیص می دهیم) اگر یک aپیدا کردیم مرحله ی قبل را دوباره اجرا می کنیم. در غیر این صورت، آنقدر راست می رویم تا ببینیم bبدون علامت ای باقی مانده است یا نه.

اگر بود، Acceptمی کنیم و در غیر این صورت Rejectمی کنیم.

این توصیف یک توصیف قانع کننده برای یک ماشین تورینگ است. همچنین چون که aها و bها را به صورت زیگ زاگی با هم متناظر کرده ایم، می فهمیم که برابری تعداد آن ها را هم به درستی چک می کنیم. در اول کار هم به کمک یک رفتار DFAمانند فرم کلی رشته را کنترل می کنیم. پس این ماشین تورینگ همان رفتاری که انتظار داریم را از خود به نمایش می گذارد و زبان آن همان Lاست. پس به خواسته ی مسئله رسیده ایم.

$L3 = \{w : n_a(w) = n_b(w)\}$

توصیف سطح بالای این ماشین تورینگ به صورت زیر است:

۱حرف اول نوار را با یک نقطه Markمی کنیم. در ادامه اگر این خانه را عوض کردیم، نقطه ی آن را نگه می داریم تا همچنان اول نوار مشخص باشد.
۲به اول نوار می رویم و از آنجا به راست حرکت می کنیم تا اولین حرف aرا ببینیم. اگر هیچ aای ندیدیم، ۳را اجرا می کنیم. اما اگر به یک aرسیدیم، آن را به Xتبدیل می کنیم و ۴را اجرا می کنیم.
۳به اول نوار می رویم و از آنجا به سمت راست حرکت می کنیم. اگر حرف bای دیدیم، رشته را Rejectمی کنیم. در غیر این صورت و اگر به خانه ی خالی رسیدیم، رشته را می پذیریم و به حالت پذیرش می رویم.
۴به اول نوار می رویم و از آنجا شروع به راست رفتن می کنیم. اگر به یک bرسیدیم، آن را به X تبدیل می کنیم و سپس ۲را اجرا می کنیم. اما اگر bندیدیم و به خانه ی خالی رسیدیم، رشته را Rejectمی کنیم.
این توصیف یک توصیف قانع کننده برای یک ماشین تورینگ است. رفتار آن به این صورت است که به ترتیب به ازای هر aبه، دنبال یک bمی گردد و این دو با هم تناظر می دهد. در صورت تناظر هردو را به Xتبدیل می کنند و عملا از نوار حذف می کند. این روند را تا آنجایی ادامه می دهد که یا a نداشته باشیم و bداشته باشیم، یا bنداشته باشیم و در حال تناظر دادن یک aباشیم، یا همه ی حرف ها را Xکرده باشیم. تنها در حالت آخر باید رشته را بپذیریم، چون aها و bها با هم در تناظر هستند و تعدادشان برابر است. پس این ماشین تورینگ همان رفتاری که انتظار داریم را از خود به نمایش می گذارد و زبان آن همان Lاست. پس به خواسته ی مسئله رسیده ایم.