دانشگاه اصفهان
دانشکده علوم ریاضی و کامپیوتر

Design and Analysis of Algorithms

طراحی و تحلیل و الگوریتم ها

محمد ملائی

عنوان:

تمرینات ۴

نیم‌سال دوم ۱۴۰۲-۱۴۰۳

نام استاد درس

جعفر الماسی زاده

تمرین ۱

این سه مسأله را (که هر سه در رده NPC هستند) در نظر بگیرید:

- **مسأله ۱**: فرض کنید A و B و C سه مجموعه مجزای n عنصری و $T \subseteq A \times B \times C$ مجموعه‌ای از سه‌تایی‌های مرتب باشد. زیرمجموعه‌ای از n سه‌تایی را (در صورت وجود) در T بیابید که هر عنصر در $A \cup B \cup C$ ، در دقیقاً یکی از آن n سه‌تایی قرار داشته باشد.

- **مسأله ۲**: با این فرض که ماتریس A یک ماتریس صفر_یک $m \times n$ و $1 = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix}$ بردار m بعدی تمام ۱ باشد، بردار مجهول صفر_یک $x = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}$ (در صورت وجود) بیابید که معادله ماتریسی $Ax = 1$ برقرار شود.

- **مسأله ۳**: مجموعه S از n عدد صحیح، و عدد صحیح d داده شده است؛ زیرمجموعه‌ای از مجموعه S را (در صورت وجود) بیابید که مجموع اعداد آن برابر با d باشد.

الف نشان دهید که مسأله ۱ را میتوان به مسأله ۲ تبدیل کرد.

ب نشان دهید که مسأله ۲ را میتوان به مسأله ۳ تبدیل کرد.

جواب

الف

ابتدا مسائل ۱ و ۲ را به صورت دقیقتر بیان می‌کنیم.
سه مجموعه $A = \{a_1, a_2, \ldots, a_n\}$ و $B = \{b_1, b_2, \ldots, b_n\}$ و $C = \{c_1, c_2, \ldots, c_n\}$ را در نظر می‌گیریم. $D = A \cup B \cup C$ که آن را از آنجایی که این سه مجموعه متمایز اند بصورت $D = \{d_1, d_2, \ldots, d_{3n}\}$ می‌نویسیم. مجموعه جواب را P می‌نامیم و آن را به صورت $P = \{p_1, p_2, \ldots, p_n\}$ تعریف می‌کنیم به طوری که p_i یک سه تایی است و $T = \{t_1, t_2, \ldots, t_m\}$، که طبق صورت سوال $P \subseteq T \subseteq A \times B \times C$.
با توجه به تعاریف بالا، بیان مسئله به شکل زیر خواهد بود:

$$\forall d \in D \; \exists! \; p \in P : d \in p$$

در مساله دوم ماتریس A را به شکل $A = [a_{ij}]; \; a_{ij} = 0, 1$ تعریف می‌کنیم و از آنجایی که به ازای بردار صفر_یک x داریم $Ax = 1$، دستگاه معادلات به شکل زیر خواهد بود:

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n &= 1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n &= 1 \\ &\vdots \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n &= 1 \end{aligned}$$

از آنجایی که ماتریس A و بردار x صفر_یک هستند، در هریک از معادلات تنها یک جمله می‌تواند یک باشد و بقیه صفر خواهند بود بنابراین بیان این مساله به شکل زیر خواهد بود:

$$\forall i \; \exists! \; j : a_{ij}x_j = 1, \; i = 1, \ldots, m, j = 1, \ldots, n$$

پس از بیان دقیق‌تر مسائل، حال باید نحوه تبدیل مسئله ۱ به مسئله ۲ را بیان کنیم.
ماتریس A را بدین صورت تعریف می‌کنیم که:

$$A = [a_{ij}]; a_{ij} = 1 \; if \; d_i \in t_j \; else \; 0$$

بدین معنا که اگر عضو i ام D در عضو j ام T باشد، آنگاه $a_{ij} = 1$ و در غیر این صورت $a_{ij} = 0$ است. حال پس از حل دستگاه، nـ بردار x، جواب مسئله اول است و ۱ بودن درایه k ام آن نشان دهنده حضور t_k در P است.

ب

فرض میکنیم A یک ماتریس $d \times n$ است و $Ax = 1$. دستگاه معادلات بصورت زیر خواهد بود:

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n &= 1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n &= 1 \\ &\vdots \\ a_{d1}x_1 + a_{d2}x_2 + \cdots + a_{dn}x_n &= 1 \end{aligned}$$

با افزودن همه‌ی معادلات به یکدیگر به معادله زیر دست می‌یابیم:

$$s_1x_1 + s_2x_2 + \cdots + s_nx_n = d$$

که در آن $s_j = \sum_{i=1}^{d} a_{ij}$ است. از آنجایی که بردار x یک بردار صفرـ یک است، بدین معنی است که پس از یافتن جواب اگر x_i یک باشد، آنگاه s_i در این مجموع محاسبه شده است و اگر صفر باشد، محاسبه نشده است. این تعبیر در واقع همان زیر مجموعه‌ای از مجموعه $S = \{s_1, s_2, s_n\}$ است که مجموع عناصر آن برابر d است.

تمرین ۲

الف تصور کنید شما که به عنوان کارشناس امنیت رایانه در شرکتی بزرگ مشغول به کار شده‌اید، به تازگی متوجه این موضوع شده‌اید که بسیاری از رایانه‌های شرکت با بدافزاری آلوده شده‌اند که باید از وبگاه‌های ناپاکی که کاربران دیده‌اند، آمده باشد. برای هر رایانه آلوده، شما یک فایل سابقه دارید که حاوی لیست تمام وبگاه‌هایی است که (از آخرین باری که رایانه برای کشف بدافزارها پویش شده است) از طریق آن رایانه دیده شده‌اند. شما با بررسی فایله‌ای سابقه، متوجه میشوید که هیچ وبگاهی وجود ندارد که از طریق همه رایانه‌ها دیده شده باشد. بنابراین، نتیجه میگیرید که بدافزار را تعدادی از وبگاه‌ها به رایانه‌ها تزریق کرده‌اند و محتمل‌ترین وبگاه‌ها، آنهایی هستند که در کوچکترین مجموعه‌ای از وبگاه‌ها که از طریق همه رایانه‌های آلوده (نه لزوماً هر یک از رایانه‌های آلوده) دیده شده باشند. آیا مسأله تعیین کمترین تعداد از وبگاه‌هایی که از طریق همه رایانه‌های آلوده دیده شده‌اند، در رده NPC است یا خیر؟

ب مجموعه $A = \{a_1, a_2 \ldots, a_n\}$ از اعداد صحیح مثبت و اعداد k و b داده شده‌اند و ما می‌خواهیم بدانیم که آیا می‌توان مجموعه A را به k زیرمجموعه دو به دو مجزای $A_1, A_2, \ldots, A_k$ تقسیم کرد به قسمی که «مجموعِ مربعاتِ مجموع اعداد زیرمجموعه‌ها» حداکثر b باشد:

$$\sum_{i=1}^{k}(\sum_{a_j \in A_i} a_j)^2 \leq b$$

آیا این مسأله، در رده NPC است یا خیر؟

جواب

الف

بله، این مسئله از رده NPC است. ابتدا این مساله را به شکل یک مسئله تصمیم گیری در می‌آوریم:

آیا مجموعه‌ای از وبگاه‌ها با شرایط مسئله وجود دارد که اندازه آن از k کمتر باشد ؟

برای نشان دادن اینکه این مسئله در رده NP است، فرض می‌کنیم مجموعه S جواب پیشنهادی مسئله است. درستی این جواب را بدین صورت بررسی می‌کنیم که هر یک از کامپیوترهای آلوده حداقل از یکی از وبسایت‌های این مجموعه بازدید کرده است در غیر اینصورت جواب پیشنهادی، جواب مسئله نخواهد بود.

حال برای نشان دادن اینکه این مسئله در رده NPC است، باید یکی از مسائل شناخته شده در این رده را به این مسئله تبدیل کنیم. برای این کار مسئله پوشش راسی را تبدیل خواهیم کرد. گراف $G =< V, E >$ را در نظر می‌گیریم. مجموعه راس‌های گراف را مجموعه‌ی همه‌ی وبسایت‌هایی که کاربران بازدید کرده‌اند در نظر می‌گیریم و آن را $W = \{w_1, w_2, \ldots, w_n\}$ می‌نامیم. یال گراف را کامپیوترهای آلوده‌ای درنظر می‌گیریم که تنها از راس‌های دوسر خود بازدید کرده‌اند بدینصورت که $C = \{\{c_{i1}, c_{i2}\} \,|(c_{i1}, c_{i2}) \in E\}$. اگر V' جواب مسئله پوشش راسی باشد، آنگاه حداقل یکی از راس‌های u یا v واقع بر هر یال $(u, v) \in E$ در مجموعه V' وجود خواهد داشت که همان جوابی است که از این مسئله می گیریم بنابراین این مسئله نیز در رده NPC قرار دارد.

ب

این مسئله نیز در رده NPC قرار دارد. بررسی درستی جواب پیشنهادی این مسئله به سادگی در زمان چند جمله‌ای انجام پذیر است پس باید یکی از مسائل این رده را به این مسئله تبدیل کنیم. مسئله افراز مجموعه را برای این کار انتخاب می‌کنیم بدین صورت که $k = 2$ و اگر مجموع اعضای A، S باشد، آنگاه $\frac{S^2}{2} = (\frac{S}{2})^2 + (\frac{S}{2})^2 = b$ خواهد بود. واضح است که جواب این مسئله همان جواب مسئله افراز خواهد بود، بنابراین این مسئله در رده NPC قرار دارد.

تمرین ۳

فروشگاهی بزرگ برای تحلیل رفتار مشتریان خود، آرایه‌ای دو بعدی A را نگهداری میکند که سطرهای آن، متناظر با مشتریان هستند و ستون‌های آن، متناظر با کالاهایی هستند که به مشتریان فروخته است. خانه $A[i,j]$ نمایانگر این است که مشتری i چند قلم از کالای j را خریده است.

این مثالی است کوچک از چنین آرایه‌ای:

	ماکارونی	کره	نوشابه	شیر
مشتری ۱	۳	۰	۶	۰
مشتری ۲	۰	۰	۳	۲
مشتری ۳	۷	۰	۰	۰

یکی از کارهایی که فروشگاه میتواند با داده‌ها انجام دهد، پیدا کردن زیرمجموعه‌ای متنوع از مشتریان است: میگوییم که یک زیرمجموعه S از مشتریان، متنوع است اگر هیچ یک از دو مشتری عضو این مجموعه، یک کالا را نخریده باشند (یعنی هر کالایی در فروشگاه را حداکثر یکی از مشتریان در مجموعه S خریده باشد). مجموعه‌های متنوع مشتریان میتوانند مفید باشند؛ مثلاً می‌توان از آنها به عنوان منبعی برای تحقیقات بازاریابی استفاده کرد.

مسأله فروشگاه را می‌توان به این شکل بیان کرد: آرایه A با ابعاد $m \times n$ (به شکلی که توصیف شد) و عدد صحیح k $(k \leq m)$ مشخص شده است؛ آیا زیرمجموعه‌ای از حداقل k مشتری وجود دارد که متنوع باشد؟

الف نشان دهید که این مسأله، در رده NP است.

ب نشان دهید که این مسأله، در رده NPC است.

جواب

الف

فرض می‌کنیم $C = \{c_1, c_2, \ldots, c_n\}$ جواب پیشنهادی مسئله باش. پس از بررسی اینکه تعداد اعضای این مجموعه بیشتر از k است، به ازای هر کالا یک بار این لیست را پیمایش کنیم و اگر کالا در بیشتر از یک لیست قرار داشت ، آن را رد و در غیر اینصورت به عنوان جواب مسئله می‌پذیریم.

ب

ابتدا مساله را بصورت دقیق‌تر توصیف می‌کنیم: اگر مجموعه همه‌ی کالاها را P بنامیم، داریم $P = \{p_1, p_2, \ldots, p_n\}$. کالاهایی که هر مشتری خریده‌است را با C_i نشان می‌دهیم به طوری که $C_i \subseteq P$. بنابراین مجموعه کالاهای خریداری شده توسط مشتری‌ها را $C = \{C_1, \ldots C_n\}$ تعریف می‌کنیم. جواب مسئله مجموعه S خواهد بود و داریم:

$$S = \{S_i \mid S_i \in C\};\ \forall S_j, S_k : S_j \cap S_k = \emptyset$$

برای اینکه نشان دهیم این مسئله در رده NPC است، می‌توانیم یکی از مسائل این رده را به این مسئله تبدیل کنیم. برای این کار مسئله مجموعه مستقل را انتخاب می‌کنیم بدین صورت که: گراف $G =< V, E >$ را در نظر می‌گیریم، جواب مسئله بزرگترین زیر مجموعه $V' \subseteq V$ از رئوس گراف است به طوری که هیچ یک از دو راس این مجموعه با یالی به هم وصل نباشند.

با پیمایش گراف G، به ازای هر راس یک مجموعه از یال‌های متصل به آن می‌سازیم بدین معنی که هر راس یک مشتری و هر یال نشان دهنده یک نوع کالا و اتصال دو راس با یک یال نشان دهنده‌ی خرید یک کالا توسط دو مشتری خواهد بود. با این توضیحات، مجموعه ساخته شده همان مجموعه C خواهد بود و حل آن، جواب مسئله مجموعه مستقل را به ما می‌دهد یعنی مجموعه S.

بنابراین این مسئله نیز در رده NPC است.

-CS.StackExchange-

تمرین ۴

شما قرار است که به یک شرکت مشاوره پزشکی که در حال طرح‌ریزی یک زمانبندی کاری برای حضور پزشکان در یک بیمارستان بزرگ است، کمک کنید. پزشکان یک زمانبندی کاری روزانه و منظم دارند که در بیشتر اوقات پاسخگوی نیازهای آنها است. با وجود این، حالا آنها لازم است به همه موارد خاص بپردازند و به ویژه، می‌خواهند از این مطمئن شوند که در هر روز تعطیلی، حداقل یک پزشک، در بیمارستان حضور کاری داشته باشد.

مسأله از این قرار است: k دوره تعطیلی وجود دارد (مثل تعطیلات نوروزی یا تعطیلات تابستانی) که هر یک از آنها چند روز متوالی طول میکشد. D_j را مجموعه روزهای j اُمین دوره تعطیلی بگیرید؛ ما به اجتماع همه این روزها، یعنی $\bigcup_j D_j$ ، مجموعه همه روزهای تعطیلی میگوییم.

n پزشک در بیمارستان کار میکنند؛ S_i را مجموعه روزهای تعطیلی بگیرید که پزشک i ، می‌تواند در آن روزها سر کار حاضر شود. (این مجموعه، ممکن است تنها شامل روزهایی خاص از یک دوره تعطیلی باشد نه همه روزهای آن دوره تعطیلی. مثلاً یک پزشک ممکن است بتواند در جمعه یا شنبه یا یکشنبه یک دوره تعطیلی در بیمارستان حاضر شود، اما در پنج شنبه آن دوره تعطیلی نتواند در بیمارستان حاضر شود.)

الف الگوریتمی کارا را توصیف کنید که این اطلاعات را بگیرد و مشخص کند که آیا تحت قیدهای زیر، میتوان حداقل یک پزشک را برای حضور در هر روز تعطیلی، انتخاب کرد یا خیر:

- هر پزشک باید در مجموع حداکثر در c روز تعطیلی کار کند، و این c روز باید فقط از روزهایی باشد که او امکان حضور در بیمارستان را دارد (مقدار c مشخص است).
- هر پزشک باید حداکثر در یکی از روزهای هر دوره تعطیلی j ، یعنی در یکی از روزهای مجموعه D_j کار کند. (به عبارت دیگر، یک پزشک خاص میتواند در چند روز تعطیلی در طول سال کار کند، اما نباید در دو روز یا بیشتر از هر دوره تعطیلی کار کند.)

الگوریتم، یا باید روزهای کاری پزشکان در تعطیلات را تحت قیدهای مذکور تعیین کند، یا آنکه به درستی این نتیجه را گزارش کند که نمی‌توان بدون نادیده گرفتن قیدها، روزهای کاری پزشکان را تعیین کرد.

ب با یک مثال، نحوه اجرای الگوریتم را توضیح دهید

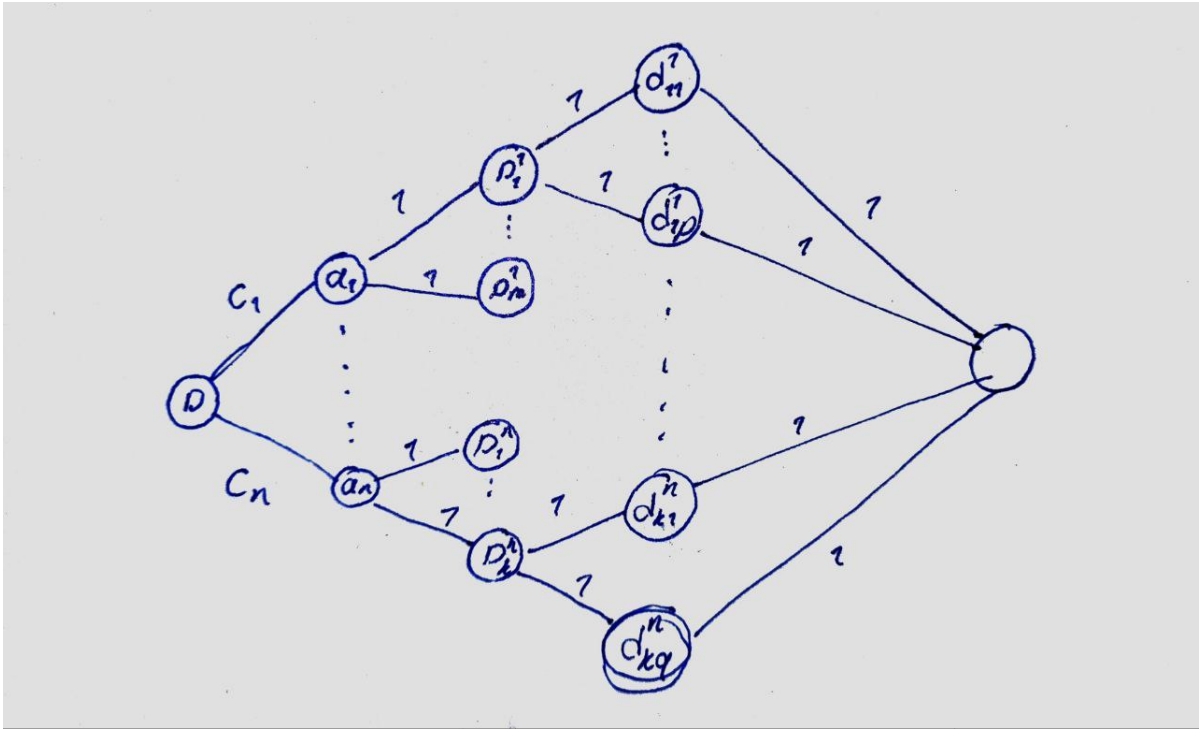
جواب

برای حل این مسئله از الگوریتم جریان بیشینه استفاده می‌کنیم بدین شرح که مسئله را به گرافی سه سطحی تبدیل می‌کنیم که در سطح اول، راس‌ها نشان‌دهنده دکترها و یال‌هایی که از مبدا به آنها وصل شده است، دارای حداکثر جریانی هستند که با c_i که حداکثر روزهایی است که دکتر i ام می‌تواند در تعطیلات به بیمارستان بیاید برابر است. در سطح دوم راس‌های سطح اول به دوره‌هایی تعطلاتی که می‌توانند در آن بازه در بیمارستان حاضر شوند متصل است و یال‌های بین این دو حداکثر، جریان ۱ را می‌توانند انتقال دهند و سطح سوم روزهایی از این دوره‌های تعطیلات است که دکتر i ام می‌تواند در آنها حاضر شود. حداکثر جریان گذرنده از یال‌های بین این راس‌ها نیز یک خواهد بود و در انتها همه‌ی این راس‌ها به راس مقصد متصل هستند و به طبع یال‌های متناظر دارای ظرفیت ۱ خواهند بود. اگر جریان ارسالی از مبدا برابر با تعداد همه‌ی روزهای تعطیل باشد و با جریان دریافتی در مقصد برابر باشد، آنگاه این مسئله جواب دارد و در غیر اینصورت جواب ندارد.

اگر مجموعه دکترها به شکل $A = \{a_1, \ldots, a_n\}$ و تعداد همه‌ی روزهای تعطیل را D بنامیم و مجموعه تعطیلاتی که دکتر i ام می‌تواند در آنها حاظر شود را D^i بنامیم داریم:

$$D^i = \{D^i_1, \ldots, D^i_m\};\ D^i_j = \{d^i_{j1}, \ldots d^i_{jp}\}$$

گراف مورد بحث به شکل زیر در خواهد آمد:



تمرین ۵

این مسأله را در نظر بگیرید: با گرفتن الفبای Σ و مجموعه S از رشته‌های ممنوعه و عدد n ، رشته‌ای را به طول n با الفبای Σ بسازید که هیچ یک از عناصر مجموعه S ، زیررشته آن نباشد.
برای مثال، اگر $\Sigma = \{0,1\}$ و $S = \{01,10\}$ و $n = 4$ باشند، آنگاه دو جواب مقبول مسأله عبارتند از 0000 و 1111 ؛ اما اگر $\Sigma = \{0,1\}$ و $S = \{0,11\}$ و $n = 4$ باشند، رشته مطلوب وجود نخواهد داشت.

الف یک الگوریتم عقبگرد برای این مسأله طراحی کنید و با دو مثال مذکور، نحوه اجرای آن را با رسم درخت فضای حالت توضیح دهید.

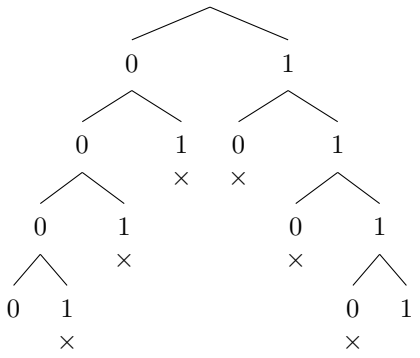
ب برنامه‌ای برای پیاده‌سازی الگوریتم بنویسید. درستی و کارایی برنامه خود را با ورودی‌های مختلف (الفباهای مختلف و مجموعه رشته‌های ممنوعه مختلف و طول‌های رشته‌های مطلوب مختلف) آزمایش کنید. هم خروجی برنامه و هم زمان اجرای برنامه در هر مورد را در جواب خود ذکر کنید.

جواب

الف

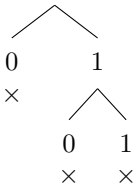
الگوریتم عقبگرد بدین صورت خواهد بود که هر گره می‌تواند به رشته قبلی هریک از اعضای Σ را اضافه کند و این بدین معنی است که هر گره از درخت به تعداد اعضای Σ فرزند دارد. الگوریتم در هر مرحله شرایط مسئله را بررسی می‌کند تا زیردرخت‌های کمتری تولید کند.
در مثال اول درخت فضای حالت به شکل زیر خواهد بود :

(1)



و درخت فضای حالت مثال دوم :

(2)



ب

```python
def generate_string(alphabet, length, forbidden, prefix=""):
    if length == 0:
        return [""]
    else:
        result = []
        for char in alphabet:
            new_prefix = prefix + char
            if not any(str in new_prefix for str in forbidden):
                for string in generate_string(
                    alphabet, length - 1, forbidden, new_prefix
                ):
                    result.append(char + string)
        return result
```

مقدار $prefix$ در این برنامه رشته‌ای است که با پیمایش از ریشه تا راس کنونی ساخته شده است و $result$ از اضافه کردن مقدار گره کنونی به هر یک رشته‌های ساخته شده توسط زیردرخت های این گره به دست می‌آید، بقیه برنامه همان چیزی است که در الگوریتم قسمت قبل توضیح داده شده است.

نتیجه آزمایش برنامه برای ورودی‌های مختلف به شرح زیر است:

```
    alphabet = 01, length = 5
    forbidden = ['001', '11']
    Generated 6 strings
    strings = ['00000', '01000', '01010', '10000', '10100', '10101']
    3.5e-05 seconds

    alphabet = 01, length = 20
    forbidden = ['0101', '1100']
    Generated 67483 strings
    0.18099 seconds

    alphabet = abc, length = 5
    forbidden = ['aa', 'acb', 'abcabc', 'b']
    Generated 13 strings
    strings= ['acaca', 'acacc', 'accac', 'accca', 'acccc', 'cacac', 'cacca', 'caccc', 'ccaca',
     'ccacc', 'cccac', 'cccca', 'ccccc']
    0.000417 seconds

    alphabet = abc, length = 25
    forbidden = ['aabb', 'b', 'ccaa']
    Generated 5600910 strings
    16.842465 seconds

    alphabet = 123456, length = 12
    forbidden = ['1', '2', '3', '45']
    Generated 121393 strings
    0.333522 seconds

    alphabet = abcdefghijklmnopqrstuvwxyz, length = 5
    forbidden = ['hello', 'ae', 'ua', 'akw']
    Generated 11741131 strings
    8.276327 seconds
```