Języki Formalne i Złożoność Obliczeniowa

Rozwiązanie zadania 40XL

Jakub Mendyk

6 kwietnia 2020

Treść

Udowodnij, że dla każdego (dostatecznie dużego) n istnieje PDFA $\mathcal{A} = \langle \Sigma, Q, q_0, F, \delta \rangle$, taki że $|Q| = n$ i że $csync(Q)$ jest niepusty ale nie zawiera słowa krótszego niż $p(n)$, gdzie p jest dowolnym, ustalonym wcześniej, wielomianem. Zakładamy, że $\Sigma = \{0, 1\}$.

Pomysł rozwiązania

Rozwiązanie dla XL rozszerza konstrukcje z poprzednich podpunktów. Zaczniemy od kilku skierowanych cykli, o długościach będących kolejnymi liczbami pierwszymi oraz dodatkowego stanu – jedynego gdzie może nastąpić (ostrożna – tylko o takiej w zadaniu myślimy więc będę to słowo pomijał) synchronizacja. Poprawimy konstrukcję dla przypadku gdy każdy każdy stan ma zostać zsynchronizowany. Ostatecznie ograniczymy się alfabetu dwuelementowego. Pokażemy, że długość najkrótszego słowa synchronizującego jest wykładniczo zależna od liczby wierzchołków. Lektura opisów konstrukcji dla L i M nie jest raczej konieczna dla zrozumienia rozwiązania XL, jednak – moim zdaniem – umożliwia prześledzenie procesu myślowego prowadzącego do konstrukcji dla podpunktu XL.

Rozwiązanie

Przeanalizujmy warunki zadania i spróbujemy je uprościć – zapisane kwantyfikatorami wyglądają następująco (może komuś – na przykład mi – pomoże w lepszym zrozumieniu treści):

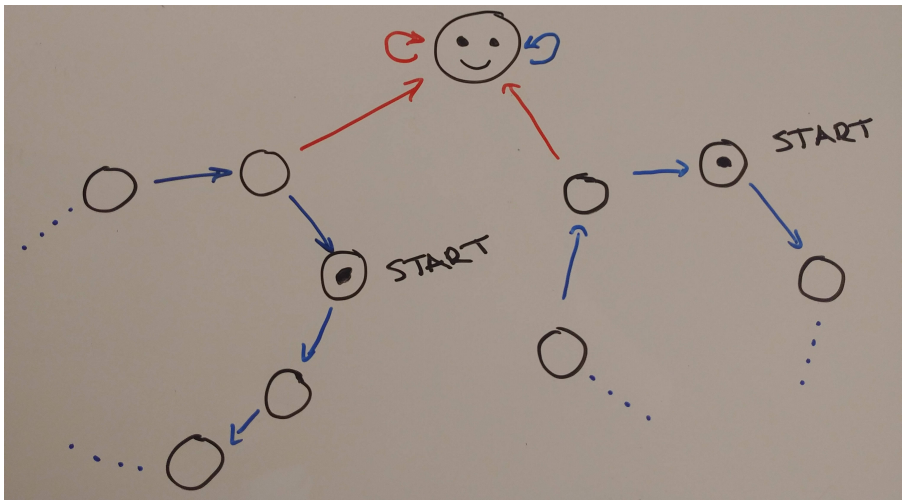
$$\forall p \text{ – wielomian } \exists N \ \forall n \geqslant N \ \exists \mathcal{A} \ \ (|Q| = n \wedge csync(Q) \neq \emptyset \wedge \forall w \in csync(Q) \ \ |w| \geqslant p(n))$$

Wiadomo, że każdy wielomian $p(n)$ można ograniczyć z góry wielomianem $p_m(n) = n^m$ dla pewnego m, a ten rośnie asymptotycznie wolniej niż jakakolwiek funkcja wykładnicza (a^n, dla pewnego $a \geq 1$).

Widząc kwantyfikatory od razu przychodzi nam na myśl walka z diabłem. Możemy myśleć o tym zadaniu, że dla danego przez diabła wielomianu oraz odpowiednio (naszym zdaniem) dużego n, potrafimy skonstruować *PDFA* $\mathcal{A}$ o n stanach, że koledzy diabła, startujący ze wszystkich wierzchołków (stanów), będą musieli zrobić co najmniej $p(n)$ kroków żeby się spotkać – nie ważne jak dobrze by im diabeł podpowiadał co mają robić.

W opisach konstrukcji, zamiast alfabetu składającego się z cyfr będę używał kolorów.

Konstrukcja dla M:



Rozważamy (uogólnioną wersję M – na potrzeby dalszych konstrukcji) synchronizację k stanów – na rysunku (ukazującym dwa wybrane cykle) są oznaczone etykietą „start". Graf stanów i przejść między nimi automatu $\mathcal{A}$ składa się z:

- cykli o niebieskich krawędziach oraz
- k czerwonych krawędzi – zaczynających się w stanie (nazwijmy go wyjściowym) poprzedzającym któryś wierzchołek „start" (będziemy go nazywać startowym), a kończących się w środkowym stanie etykietowanym, nie bez powodu, uśmiechniętą buźką.
- tegoż stanu „:-)", mającego przejścia obu kolorów do samego siebie.

Zauważmy, że jest to jedyny stan, do którego można stany startowe zsynchronizować – nie ma bowiem ścieżek między cyklami. Zatem słowa synchronizujące muszą mieć postać $\boldsymbol{n^l z}$ dla pewnych l.

Niech $\mathbb{P}_k$ oznacza k pierwszych liczb pierwszych – będą to długości cykli. Słowo synchronizujące musi przeprowadzić stany startowe, tak aby jednocześnie przeszły na stan z krawędzią prowadzącą do „:-)"– inaczej niemożliwe jest przejście czerwoną krawędzią i zsynchronizowanie stanów.

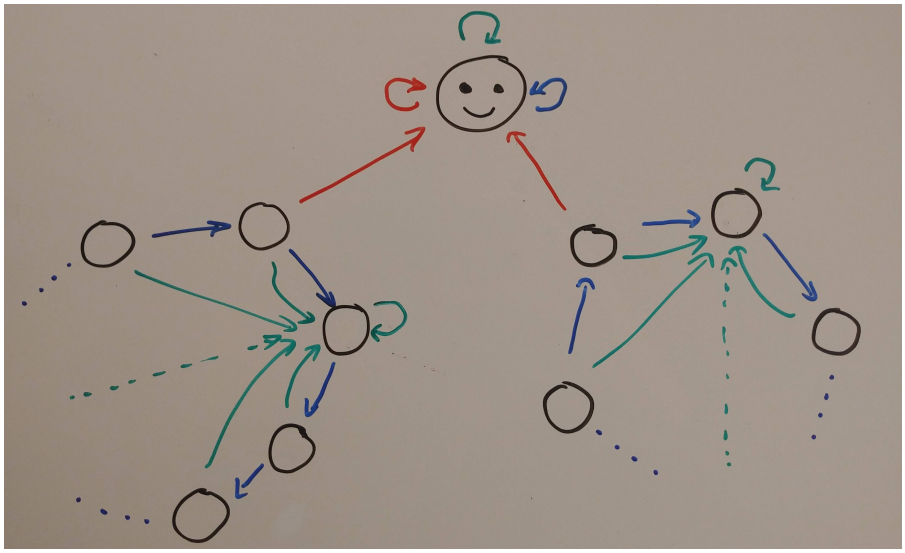
Musimy znaleźć takie l, że:

$$l \equiv p_i - 1 \pmod{p_i}$$

Łatwo zauważyć, że najmniejsze l, będące rozwiązaniem tego układu to $l = NWW(\mathbb{P}_k) - 1$, długości cykli są liczbami pierwszymi więc $l = \prod_{i=1}^{k} p_i - 1$. Stąd najkrótsze słowo, które synchronizuje stany początkowe to $\boldsymbol{n^l z}$, które ma długość $\prod_{i=1}^{k} p_i$. Wartość ta ma wzrost wykładniczy względem n, czujemy więc że zmierzamy w dobrym kierunku.

Konstrukcja dla L:

Zamiast synchronizować k stanów automatu, musimy zsynchronizować wszystkie. Z każdego wierzchołka w każdym cyklu prowadzimy krawędź zieloną do startowego.



Słowa synchronizujące muszą przed wystąpieniem z, zawierać z - w przeciwnym razie nie byłoby możliwe wczytanie z ze wszystkich stanów i synchronizacja. Zsynchronizowanie wszystkich stanów w cyklu nazwiemy *wstępnym zsynchronizowaniem*.

Jeżeli $n^{l'}zn^lz$ synchronizuje Q, to zn^lz także – niezależnie od wartości l', wczytanie z spowoduje zsynchronizowanie stanów każdego cyklu do stanu startowego. Obecność n przed z jest bez znaczenia, stąd zn^lz jest krótszym słowem w $csync(Q)$.

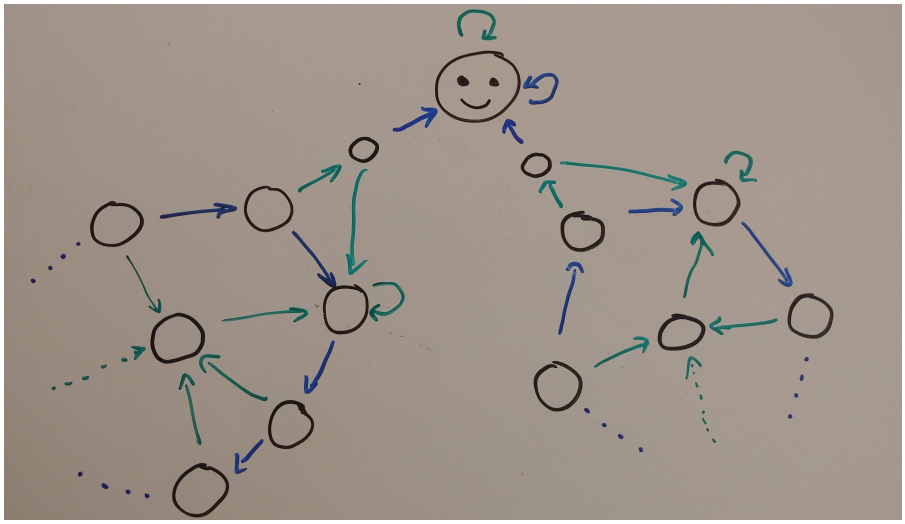
Po wstępnym zsynchronizowaniu otrzymujemy sytuację analogiczną do tej z M, więc najkrótsze słowo w $csync(Q)$ to $zn^{NWW(\mathbb{P}_k)}z$ o długości $\prod_{i=1}^{k} p_i + 1$.

Konstrukcja dla XL:

Nie mamy już do dyspozycji trzy-, a jedynie dwuelementowy alfabet. Poradzimy sobie z tym zmieniając konstrukcję cykli.

Właściwe cykle nadal mają długości będące liczbami pierwszymi. „Dodajemy” dwa nowe stany:

- środkowy – do którego wchodzą zielone krawędzie ze wszystkich stanów na cyklu prócz stanu wyjściowego oraz startowego,
- stan będący rozbiciem niegdyś czerwonej krawędzi (nazwiemy go przejściowym) – ma krawędzie:
  - zieloną ze stanu wyjściowego,
  - niebieską do „:-)”,
  - zieloną do stanu startowego.



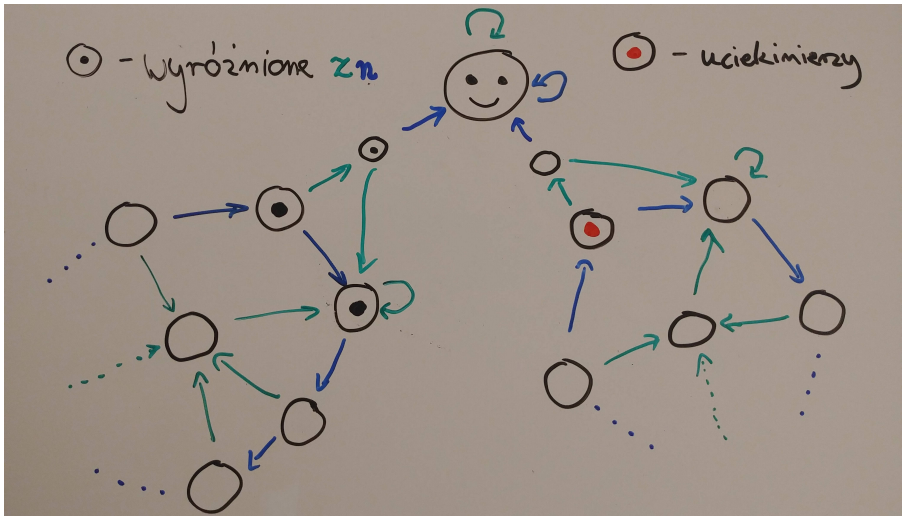
Z cyklu można wyjść już nie tylko, gdy robią to jednocześnie koledzy diabła ze wszystkich cykli. Zobaczymy jednak, że najkrótsze słowo synchronizujące nadal będzie wystarczająco długie. *Próbą ucieczki* nazwiemy sytuację, w której po wczytaniu pewnej liczby znaków, w którymś cyklu stany przejdą na stan wyjściowy.

Zauważmy, że stan środkowy ma tylko jedną krawędź – zieloną – czyli słowo synchronizujące musi zaczynać się od $\boldsymbol{z}$. Po wykonaniu pierwszego ruchu wszystkie, prócz dwóch, stany na cyklu przejdą na środkowy, ponownie musimy przejść zieloną krawędzią. Po wczytaniu $\boldsymbol{zz}$ wszystkie stany na cyklu (oraz środkowy, wyjściowy i przejściowy) przejdą na startowy – mamy wstępną synchronizację stanów.

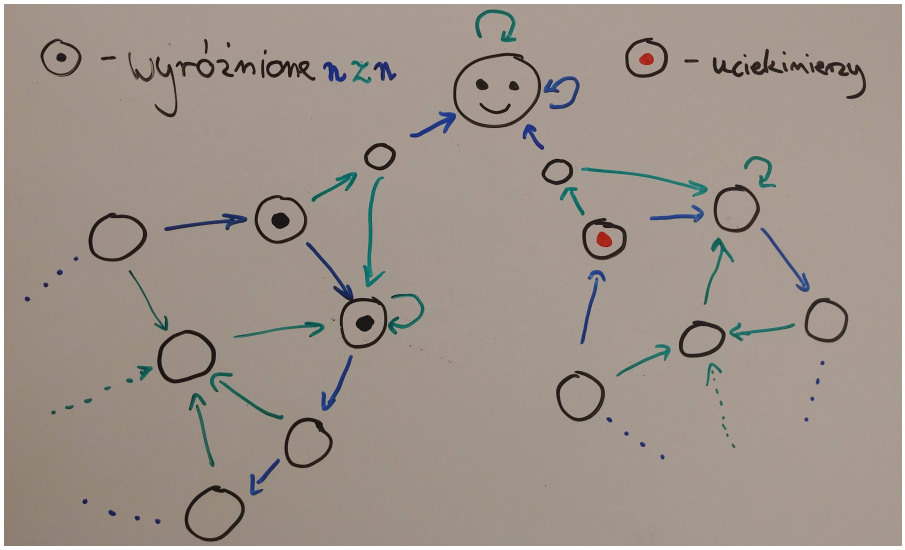
Jak zauważyliśmy, koledzy diabła mogą próbować uciec ze swojego cyklu nie robiąc tego jednocześnie wraz z kolegami z innych cykli. Niezmiennie jednak, mają możliwość zsynchronizowania się w stary sposób – po przejściu $NWW(\mathbb{P}_k)$ razy niebieskimi krawędziami (od chwili wstępnej synchronizacji) wszyscy znajdą się w stanach wyjściowych i jednocześnie przejdą przez stan przejściowy do stanu z uśmiechniętą buźką – $csync(Q) \neq \emptyset$.

Diabełkom nie opłaca się po wstępnej synchronizacji przechodzić zielonymi ścieżkami jeśli nie jest to próba ucieczki – wtedy diabełki są w którymś ze stanów na cyklu nie będącym wyjściowym. Jeśli grupa diabełków w którymś cyklu jest w stanie startowym, to ponownie w nim będą po przejściu zieloną krawędzią. W przeciwnym razie znajdą się w stanie środkowym – wtedy ponownie musi nastąpić przejście zieloną krawędzią. W obu przypadkach wszystkie grupy diabełków jednocześnie wylądują w stanie startowym – jak po wstępnej synchronizacji, zatem nie opłaca się im chodzić po zielonych krawędziach jeśli nie jest to próba ucieczki którejś grupki. Stąd najkrótsze słowo po przeczytaniu którego pierwsi uciekinierzy trafią do „:-)" musi być postaci $\boldsymbol{zzn^lzn}$ dla pewnego l.

By którakolwiek grupka kolegów diabła opuściła cykl, musi znaleźć się w stanie wyjściowym, po czym przejść sekwencją krawędzi $\boldsymbol{zn}$ – nie z każdego stanu można taką sekwencję wykonać! Na rysunku poniżej czarnym kolorem zostały wyróżnione stany, z których taka sekwencja jest możliwa.



Zauważmy, że do stanu wyjściowego prowadzi jedynie krawędź niebieska, więc zanim nasi uciekinierzy znaleźli się w tym stanie, musieli przejść krawędzią niebieską. Poniżej kolorem czarnym jest wyróżniony podzbiór stanów które były zaznaczone na poprzednim grafie i które posiadają niebieską krawędź wchodzącą.



Otrzymujemy, że w każdym cyklu są tylko dwa możliwe stany, w których muszą być diabełki aby co najmniej jednej grupce – będącej w stanie wyjściowym – udało się uciec. Licząc od momentu, gdy w każdym cyklu nastąpiła wstępna synchronizacja, ucieczka będzie możliwa gdy liczba wykonanych kroków l spełnia:

$$\forall i \leqslant k \quad l \equiv 0 \ (mod\ p_i) \vee l \equiv p_i - 1 \ (mod\ p_i)$$

Pierwszy warunek koniunkcji jest spełniony gdy w i-tym cyklu diabełki są w stanie startowym, drugi gdy są w wyjściowym. Przenosząc w drugim warunku 1 na lewą stronę otrzymujemy:

$$\forall i \leqslant k \quad l \equiv 0 \ (mod\ p_i) \vee l + 1 \equiv 0 \ (mod\ p_i)$$

zatem:

$$\forall i \leqslant k \quad p_i \mid l(l+1)$$

Dla każdej liczby p_i, gdzie $i \leqslant k$, p_i dzieli l lub $l+1$, a zatem p_i jest czynnikiem $l(l+1)$, stąd:

$$l(l+1) \geqslant NWW(\mathbb{P}_k)$$
$$l^2 + l - NWW(\mathbb{P}_k) \geqslant 0$$
$$l \leqslant \frac{-1-\sqrt{1+4NWW(\mathbb{P}_k)}}{2} \vee l \geqslant \frac{-1+\sqrt{1+4NWW(\mathbb{P}_k)}}{2}$$
$$l \geqslant \frac{-1+\sqrt{1+4NWW(\mathbb{P}_k)}}{2} \quad \text{(l naturalna)}$$
$$l \geqslant \sqrt{NWW(\mathbb{P}_k)} - 1$$

Nie wymusiliśmy na diabełkach jednoczesnego opuszczenia cykli, jednak wiemy że zanim któraś grupa to zrobi, nastąpi przeczytanie słowa o długości co najmniej $\sqrt{NWW(\mathbb{P}_k)}$. Wcześniej sprawdziliśmy, że $csync(Q) \neq \emptyset$.

Na tym kończy się opis konstrukcji automatu. Intuicyjnie widzimy, że spełnia ona warunki zadania. Jednak wykonajmy obliczenia by mieć pewność poprawności tej konstrukcji.

Obliczenia

Uwaga: Można zauważyć, że liczba stanów w opisanej konstrukcji nie musi sumować się (dokładnie) do $|Q| = n$. Brakujące stany dodamy „przy" stanie „:-)" – każdy z zieloną krawędzią (wiemy że wszystkie słowa synchronizujące zaczynają się z) do stanu z uśmiechniętą buźką. W taki sposób spełniamy wymóg liczby stanów nie wpływając na poprawność wcześniejszego rozumowania.

Liczba cykli wynosi k, a ich długości to k pierwszych liczb pierwszych. Do każdego cyklu mamy dwa dodatkowe stany – środkowy i przejściowy – no i mamy jeszcze stan „:-)". Wiemy, że słowa synchronizujące wszystkie stany Q mają długości wynoszące co najmniej $\sqrt{NWW(\mathbb{P}_k)}$. Skorzystamy ze wskazówki do zadania, że suma k pierwszych liczb pierwszych jest zawsze mniejsza niż $k^2 \log k$, zaś iloczyn zawsze większy niż $2^{k \log k}$.

Pokażemy, że istnieje k spełniające układ równań (pamiętamy, że zastąpiliśmy dowolny wielomian większym postaci n^m dla pewnego m):

$$\left(\sum_{i=1}^{k} p_i\right) + 2k + 1 \leqslant k^2 \log k + 2k + 1 \leqslant n$$
$$\sqrt{NWW(\mathbb{P}_k)} \geqslant \sqrt{2^{k \log k}} \geqslant 2^{(k \log k)/2} \geqslant n^m$$

Łącząc obie nierówności, wystarczy pokazać iż dla ustalonego m istnieje k, dla którego zachodzi:

$$2^{(k \log k)/2} \geqslant n^m \geqslant \left(k^2 \log k + 2k + 1\right)^m$$
$$2^{(k \log k)/2} \geqslant \left(k^2 \log k + 2k + 1\right)^m$$

Jak łatwo zauważyć, nierówność ta jest prawdziwa dla odpowiednio dużych k. Zatem, z dowolności wyboru kiedy n uznajemy za dostatecznie duże, zawsze znajdziemy takie N, że dla wszystkich $n \geqslant N$ powyższa nierówność jest prawdziwa, stąd nasza konstrukcja spełnia wszystkie wymagania zadania.