Tile Covers

3 listopada 2021

1 Introduction

1.1 Podstawy 1D

Lemat 1. Unarny tekst $n \times m$ może być pokryty przez (unarnego) covera 1D długości l jeśli l dzieli n lub m (czyli nie wystarczy, że dzieli iloczyn).

Dowód. Kolorujemy elementy tablicy $n \times m$ na l kolorów - k-ty wiersz cyklicznie zaczynając od elementu $k \bmod l$ ($(k-1) \bmod l + 1$ jeśli chcemy numerować od 1).

Każdy prostokąt tablicy o wymiarach $1 \times l$ lub $l \times 1$ zawiera po jednym wystąpieniu każdego koloru, więc pokrycie jest możliwe tylko jeśli każdy kolor występuje tyle samo razy.

Liczność wszystkich kolorów w pasku $k \times l$ czy $l \times k$ jest taka sama, więc możemy zredukować problem do liczenia kolorów dla wymiarów $m, n < l$. W tym wypadku jeśli $m \neq 0, n \neq 0$, to kolor m występuje $min(n, m) > (n \cdot m)/l$ razy, więc pokrycie jest niemożliwe. Jeśli l dzieli n (m), to do pokrycia całego tekstu wystarczają same wystąpienia pionowe (poziome) - trywialne pokrycie. □

Jeśli cały tekst nie jest unarny, zaś kadydat jest, to nie może on być coverem. Dalej zakładamy, że kandydat nie jest unarny. Załóżmy, że pierwsza jego litera to a, zaś pierwsza litera różna od a to b

Jeśli pierwszy wiersz tekstu zawiera literę b (czy tam literę inną niż a), to może ona zostać pokryta wyłącznie przez poziome jego wystąpienie, tak więc pierwsze wystąpienie kandydata w pierwszym wierszu (najłatwiej wykrywane pierwszym wystąpieniem litery b) jest ściśle określone. Jednocześnie wszystkie litery przed jego początkiem muszą zostać pokryte przez wystąpienia pionowe.

Algorytm zachłanny:

Pokrywamy tekst od góry do dołu, od lewej do prawej.

Bierzemy pierwszy niepokryty element tekstu w kolejności czytania, szukamy pierwszego poziomego wystąpienia kandydata na prawo od niego w jego wierszu, który nie zawiera pokrytej pozycji (równoważnie szukamy pierwszego niepokrytego b). Pokrywamy ten fragment poziomo, zaś we wszystkich niepokrytych literach przed jego początkiem zaczynamy wystąpienie pionowe (może być tak, że nie ma w tym wierszu już wystąpienia poziomego, wtedy pokrywamy pionowymi wystąpieniami pozycje do jego końca).

Jeśli w dowolnym miejscu algorytmu natrafimy na mismatch (w tym wyjście poza tablicę), to kandydat nie może być coverem. Jeśli dojdziemy do samego końca bez takiego problemu, to kandydat jest coverem.

Twierdzenie 1. Algorytm zachłanny sprawdza konkretnego kandydata w czasie $O(N)$.

Wniosek 1. Tekst może być pokryty konkretnym coverem tylko na jeden sposób.

Obserwacja 1. Dla kandydatów jednowymiarowych algorytm zachłanny zadziała również w przypadku unarnym (przy czym tutaj musimy mówić o pierwszym poziomym wystąpieniu kandydata a nie o wystąpieniu litery b).

Fakt 1. Dla dowolnego k istnieje rosnący ciąg liczb n_l, dla którego liczba dzielników n_l jest $\Omega(\log^k n_l)$ (można wziąć $P = \prod_1^k p_i$, gdzie p_i to i-ta liczba pierwsza, oraz $n_l = P^l$, wtedy liczba dzielników n_l to $(l+1)^k$ - to wymyśliłem tak na szybko).

Dla dowolnego ε dla dostatecznie dużych n liczba dzielników jest $O(n^\varepsilon)$ (tutaj znalazłem sformułowanie twierdzenia w internecie, ale nie widzę prostego dowodu).

Wniosek 2. Algorytm zachłanny znajduje wszystkie covery 1D tekstu w czasie $N^{1+o(1)}$.

Kandydaci z jednej grupy (prefiksy pierwszego wiersza, lub pierwszej kolumny) są prefiksami dłuższych kandydatów z tej samej grupy.

Przyśpieszony algorytm zachłanny:

W preprocessingu obliczamy tablicę PREF (najdłuższy prefiks kandydata zaczynający się na konkretnej pozycji w słowie) dla najdłuższego kandydata z grupy - to jest pierwszego wiersza lub pierwszej kolumny (właściwie to później jest zauważone, że co najwyżej jedna grupa ma w ogóle sens) oraz każdego wiersza i każdej kolumny.

W trakcie działania algorytmu zamiast sprawdzać czy na pozycji występuje kandydat w czasie liniowym robimy to w czasie stałym korzystając ze struktury.

Aby zapamiętać które pozycje w wierszu zostały już pokryte przy pomocy pionowego wystąpienia kandydata (które zaczyna się wyżej) posiadamy strukturę zapamiętującą te pozycje w postaci przedziałów i aktualizujemy ją dynamicznie przechodząc do kolejnego wiersza. Jeśli odległość między dwoma sąsiednimi przedziałami jest pomiędzy 1 a l (długość kandydata), dla opuszczanego wiersza, to kandydat nie może być coverem (nie pokryliśmy pionowo, a poziome wystąpienie się nie zmieści), tak więc przedziałów może być co najwyżej m/l.

Twierdzenie 2. Po preprocessingu w czasie $O(N)$ przyśpieszony algorytm zachłanny sprawdza pojedynczego kandydata w czasie $O(N/l)$.

Wniosek 3. Jako, że suma odwrotności dzielników N jest $O(\log\log N)$, algorytm ten znajduje wszystkie covery 1D w czasie $O(N \log\log N)$.

1.2 Bardziej zaawansowane 1D

Fakt 2. Cover musi być borderem jednego (z czterech) ze słów $S_1\$S_2$, gdzie \$ jest znakiem spoza alfabetu, S_1 to pierwszy wiersz lub pierwsza kolumna, zaś S_2 to ostatni wiersz lub ostatnia kolumna.

Ponadto (w przypadku nie-unarnym) co najwyżej jedna z tych kombinacji może generować covery i można to wykryć nie patrząc na resztę tekstu (może poza sprawdzeniem, czy jest on unarny).

Definicja 1. (W tej sekcji) Kandydatem dla tekstu dwuwymiarowego T nazywamy słowo jednowymiarowe U, dla którego:

- U jest borderem odpowiedniej kombinacji pierwszy (wiersz/kolumna), ostatni (wiersz/kolumna) (jedynej której bordery mogą być coverami tekstu T)
- Stosunek ilości liter każdego typu dla U jest taki sam jak dla T
- Ten odpowiedni pierwszy (wiersz/kolumna) może być pokryty bez przecięć wystąpieniami słowa U oraz literami a

Fakt 3. Jeśli słowo U jest kandydatem/coverem, to słowo pierwotne U_1, takie, że $U = U_1^k$ dla pewnego $k > 0$, również jest kandydatem/coverem.

Lemat 2. Weźmy dwóch kandydatów U, V, takich że U jest pierwotne, oraz $|U| < |V|$. Wtedy V musi być potęgą U.

Dowód. Załóżmy, że ódpowiedni"jest pierwszy wiersz. Wiemy, że U jest borderem V, oraz że pierwsze wystąpienie V w pierwszym wierszu składa się z rozłącznych wystąpień słowa U, wolnych liter a plus sufiksu długości $< |U|$, który jest prefixem U (być może pustym).

Wiemy więc, że ostatnie $|U|$ liter V tworzy z jednej strony słowo $U = U'WU''$ a z drugiej $U''a^lU'$ (dowolny z tych bloków może być pusty).

Dostajemy równanie na słowach $U'WU'' = U''a^lU'$, z którego otrzymujemy natychmiast $W = a^l$ (liczność liter a po obu stronach równania)

Jest ono równoważne równaniu

$(U'a^l)(U''a^l) = (U''a^l)(U'a^l)$, czyli $xy = yx$, z którego otrzymujemy

$U = (Xa^l)^pX, U' = (Xa^l)^qX, U'' = (Xa^l)^rX$, gdzie $p = q + r + 1$.

Teraz:

załóżmy, że X składa się z liter innych niż a (w przypadku unarnym lemat trywialnie prawdziwy) załóżmy też, że V składa się z s pełnych wystąpień U, t wystąpień wolnych liter a oraz suffixu $a^l U'$,

ponieważ stosunek liczby liter a do pozostałych w słowach a oraz $a^l U'$ może być tylko większy niż w słowie U jedyną możliwością aby ten stosunek w słowach U i V był taki sam jest to, żeby $s = l = 0$. To jednak oznacza, że słowo V jest potęgą słowa U (skoro U jest pierwotne, to $X = U$). □

Wniosek 4. Istnieje tylko jedna grupa kandydatów (całkowite potęgi jednego słowa pierwotnego).

Wniosek 5. To samo dotyczy coverów, jako że cover jest kandydatem.

Wniosek 6. To słowo pierwotne może zostać wyznaczone w czasie $O(N)$.

Dowód. Algorytm:

1. Wyznaczamy stosunek liczby wszystkich liter w tekście T.
2. Wyznaczamy która para pierwszy (wiersz/kolumna), ostatni (wiersz/kolumna) jako jedyna może zawierać kandydatów i tworzymy z nich słowo przedzielone literą spoza alfabetu
3. liczymy pierwotne bordery utworzonego słowa
4. idąc od początku słowa zliczamy ilość każdej litery (wystarczy ilość liter a i liter "nie a") i kiedy dochodzimy do długości będącej długością bordera porównujemy stosunek ilości liter a do tego w całym tekście.
5. sprawdzamy którzy z tych "prawie pokrywają" wybrany wiersz/kolumnę.

Czas $O(N + (n+m) \cdot \min(n, m))$ - najdroższym krokiem jest policzenie stosunku liter w tekście, czyli wczytanie inputu. □

Wniosek 7. Najkrótszy cover, oraz to czy w ogóle cover istnieje potrafimy wyznaczyć w czasie $O(N)$.

Twierdzenie 3. Potrafimy wyznaczyć wszystkie covery w czasie $O(N)$.

Dowód. Algorytm:

1. sprawdzamy, czy najkrótszy element grupy kandydatów pokrywa tekst T, jeżeli tak, to zapamiętujemy to pokrycie w postaci rozłącznych prostokątów pokrywających tekst (ponieważ cover ten jest nieunarny, to pokrycie jest jednoznaczne, oraz prostokąty są jednoznacznie poziome lub pionowe)
2. jeśli dwa prostokąty stykają się krótszymi bokami, to łączymy je w jeden większy (możemy to zrobić w dowolnej kolejności - zawsze wyjdzie to samo)
3. zliczamy długości powstałych prostokątów
4. potęgi najkrótszego covera, które również są coverami, to te, których długości dzielą długości każdego z otrzymanych prostokątów, czyli wystarczy policzyć największy wspólny dzielnik długości prostokątów i zwrócić wszystkie jego dzielniki będące wielokrotnościami długości najkrótszego covera.

□

1.3 Przypadek $d \times l$ $(d < l)$

Lemat 3. Tekst unarny $n \times m$ może być pokryty przez (unarnego) covera $d \times l$ w dwóch przypadkach:
(a) d dzieli długość jednego boku tekstu, zaś l drugiego
(b) $nww(d, l)$ dzieli długość jednego boku tekstu, zaś długość drugiego boku jest postaci $a \cdot d + b \cdot l$, dla całkowitych $a, b \geq 0$.

Dowód. Z przypadku jednowymiarowego wynika, że zarówno d jak i l muszą dzielić co najmniej jeden z boków (prostokąt $d \times l$ można rozłożyć na prostokąty $1 \times d$ lub $1 \times l$, co tylko ułatwiłoby pokrycie).

Jeśli zachodzi przypadek (a), to istnieje oczywisty układ prostokątów $d \times l$, którym można pokryć prostokąt $n \times m$. Załóżmy więc dalej, że $nww(d, l)$ dzieli m.

Jeśli n można wyrazić w postaci $a \cdot d + b \cdot l$, to pokrycie otrzymujemy poprzez położenie a warstw poziomo i b warstw pionowo (dzielimy tekst według wysokości na dwie części - $a \cdot d$ i $b \cdot l$, pierwszą pokrywamy używając wyłącznie wystąpień poziomych, zaś drugą używając wyłącznie wystąpień pionowych).

Z drugiej strony. Jeżeli istnieje cover, to jego wystąpienia dzielą tę kolumnę, na rozłączne odcinki długości d oraz l, a więc n musi być wyrażalne w powyższej postaci. □

Lemat 4. Macierz $d \times l$, w której: macierze $d \times d$ - skrajnie lewa i skrajnie prawa są symetryczne, oraz która posiada okres poziomy d rozbija się na symetryczne macierze $nwd(d, l) \times nwd(d, l)$. Co więcej wiersz $i + nwd(d, l)$ jest cyklicznym przesunięciem wiersza i o $nwd(d, l)$ pozycji.

Dowód. Niech $1 < z = nwd(d, l) < d$.

Z symetrii skrajnie lewej podmacierzy $d \times d$ wynika, że podmacierze $z \times z$ na jej przekątnej są symetryczne. Analogiczna własność zachodzi dla podmacierzy na przekątnej skrajnie prawej podmacierzy $d \times d$. Z poziomej okresowości te macierze $z \times z$ występują również w skrajnie lewej macierzy $d \times d$, jednak (ponieważ $z \neq d$) nie na głównej przekątnej (tutaj przez pozostałe przekątne mam na myśli główną przekątną przesuniętą cyklicznie w prawo). Korzystając kolejny raz z symetrii skrajnie lewej macierzy $d \times d$ pokazujemy tą własność dla kolejnej przekątnej (symetrycznej do tej poprzedniej). Korzystając z poziomej okresowości przechodzimy do skrajnie prawej podmacierzy i tak dalej, aż pokażemy tą własność dla wszystkich podmacierzy $z \times z$ (l/z oraz d/z są względnie pierwsze, więc nie zapętlimy się przed przejściem przez wszystkie przekątne).

Dla $z = 1$ lub $z = d$ lemat jest trywialny.

Własność ćo więcej"wynika z tego, że macierze na pozostałych przekątnych są cyklicznie przesuniętymi macierzami na głównej przekątnej (przy czym to jest bardziej taka ciekawostka, bo aktualnie nie jest to używane w algorytmie). □

Twierdzenie 4. W przypadku coverów $d \times l$ algorytm zachłanny również działa.

Dowód. Załóżmy, że wymiary kandydata oraz tekstu spełniają własność określoną w lemacie 3.

Rozważmy przypadki:

0. Wszystko jest unarne - już rozpatrzony w lemacie 3.

1. Przypadek pseudounarny - tekst podzielony na fragmenty $nwd(d, l)$ składa się z samych fragmentów symetrycznych - możemy zastosować redukcję - zamianę bloków $nwd(d, l) \times nwd(d, l)$ na pojedyncze litery (robimy to w czasie liniowym od wielkości tekstu).

2. Kandydat jest unarny, zaś tekst nie, lub kandydat składa się z symetrycznych bloków $nwd(d, l) \times nwd(d, l)$, zaś tekst nie - nie ma pokrycia.

3. Macierz utworzona z pierwszych $d \times d$ elementów kandydata nie jest symetryczna - algorytm zachłanny działa - kiedy staramy się pokryć pierwszą leksykograficznie jeszcze nie pokrytą pozycję, to podmacierz $d \times d$ z lewym górnym rogiem zawierającym tą pozycję jednoznacznie wyznacza, czy możemy zacząć tu wystąpienie poziome, czy pionowe (lub od razu odrzuca kandydata).

   Analogicznie postępujemy gdy ostatnia macierz $d \times d$ nie jest symetryczne (tylko pokrywamy od dolnego prawego rogu).

4. Kandydat nie ma okresu poziomego d (po podziale na macierze $d \times d$ (i jedną $d \times l \bmod d$) nie wszystkie są równe pierwszej = "takie wystąpienie litery innej niż a").

   Ten przypadek działa analogicznie do przypadku jednowymiarowego - pierwsze wystąpienie macierzy różnej od tej pierwszej $d \times d$ i zaczynającej się na pozycji będącej wielokrotnością d w pierwszym niepokrytym wierszu implikuje pierwsze poziome wystąpienie kandydata.

5. Kandydat ma okres poziomy d.

   Jeśli $nwd(d, l) > 1$, to korzystając z lematu 4 zachodzi przypadek 1 lub 2. Załóżmy więc, że $nwd(d, l) = 1$.

   Zauważmy, że jeśli chcemy zacząć wystąpienie (poziome lub pionowe) po poziomym wystąpieniu kandydata (w tym samym wierszu), to następuje złamanie okresu (w przeciwnym przypadku kandydat miałby okresy d oraz $l - d$, a więc i $nwd(d, l) = 1$, a więc byłby unarny - przypadek 0 lub 2).

   Tak więc tutaj również działa algorytm zachłanny, z tą różnicą, że zamiast szukać "pierwszej litery różnej od a" (bloku $d \times d$ różnego od pierwszego) szukamy miejsca w którym łamie się okres d (lub np. mamy coś już pokryte pionowo).

□

Obserwacja 2. W przypadku 2D każdy z przedstawionych algorytmów zachłannych może zadziałać błędnie dla kandydata unarnego (w przeciwieństwie do przypadku 1D).

Obserwacja 3. W przypadku 3 (pierwsza macierz $d \times d$ niesymetryczna) zachłanny algorytm stworzony dla przypadku 4 (macierz $d \times d$ symetryczna, ale kandydat nie ma okresu d) zadziała równie dobrze, przez co dodatkowy algorytm zachłanny dla tego przypadku nie jest potrzebny.

Co więcej algorytmy dla przypadków 4 i 5 różnią się tylko sposobem znalezienia pierwszego wystąpienia poziomego (na prawo od miejsca w danym wierszu), zaś pozostałe rzeczy (kryterium znalezienia/odrzucenia kandydata, znajdz pierwsze poziome wystąpienie, a to co przed nim pokryj pionowo, spamiętuj to co zostało pokryte wystąpieniem, które zaczynało się wyżej) zostają bez zmian.

Przykład 1.

a	b	a	b	a	a	b	
b	a	b	a	b	b	a	
a	b	a	b	a	a	b	
a	b	b	a	b	a	b	
b	a	a	b	a	b	a	
a	b	b	a	b	a	b	

Przykład na użycie algorytmu zachłannego z łamaniem okresu - pierwsze użyte wystąpienie covera w pierwszym wierszu występuje dopiero przed miejscem złamania okresu 2, pomimo tego, że cover występuje w tym wierszu też już wcześniej.

Inna sytuacja występuje przy znalezieniu ostatniego poziomego wystąpienia, które zostało użyte - w przedostatnim wierszu okres nie łamie się (po pozycji 2, gdzie kończy się wystąpienie pionowe), zaś koniec wystąpienia poziomego wymuszony jest przez wystąpienie pionowe, które zaczęło się we wcześniejszym wierszu.