Jakub Radoszewski, Wojciech Rytter
Treść zadania, Opracowanie

Przemysław Horban
Program

Dostępna pamięć: 256 MB.

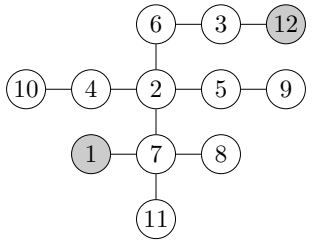
OI, etap I, 15.10–12.11.2012

Multidrink

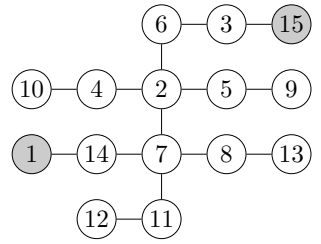
Bajtazar mieszka w Bajtowie, które słynie z tego, że przy każdym skrzyżowaniu ulic znajduje się bar mleczny. Pewnego dnia Bajtazar postanowił odwiedzić w celu tzw. „mlecznego multidrinka" wszystkie bary, każdy dokładnie raz. Bajtazar chciałby tak zaplanować trasę, żeby każdy kolejny bar był nie dalej niż dwa skrzyżowania od poprzedniego.

Skrzyżowania w Bajtowie są ponumerowane od 1 do n. Wszystkie ulice są dwukierunkowe. Między każdymi dwoma różnymi skrzyżowaniami jest tylko jedna trasa, na której żadne skrzyżowanie nie powtarza się. Bajtazar startuje przy skrzyżowaniu numer 1 i kończy przy skrzyżowaniu numer n.

Twoim zadaniem jest wyznaczenie jakiejkolwiek trasy spełniającej wymagania Bajtazara lub wypisanie informacji o braku takiej trasy.



Trasą spełniającą warunki zadania jest na przykład: $1, 11, 8, 7, 5, 9, 2, 10, 4, 6, 3, 12$.



Nie ma żadnej trasy spełniającej warunki zadania.

Wejście

W pierwszym wierszu standardowego wejścia znajduje się jedna liczba całkowita n ($2 \leqslant n \leqslant 500\,000$) oznaczająca liczbę skrzyżowań w Bajtowie. Każdy z kolejnych $n-1$ wierszy zawiera dwie różne liczby całkowite a_i oraz b_i ($1 \leqslant a_i, b_i \leqslant n$), oddzielone pojedynczym odstępem, reprezentujące ulicę łączącą skrzyżowania o numerach a_i i b_i.

W testach wartych łącznie 65% punktów zachodzi dodatkowy warunek $n \leqslant 5000$.

Wyjście

Jeśli nie istnieje żadna trasa spełniająca wymagania Bajtazara, w pierwszym i jedynym wierszu standardowego wyjścia Twój program powinien wypisać jedno słowo `BRAK`*. W przeciwnym przypadku Twój program powinien wypisać n wierszy, z których i-ty powinien zawierać numer i-tego skrzyżowania na przykładowej trasie spełniającej warunki Bajtazara. W pierwszym wierszu powinna znaleźć się liczba 1, a w n-tym wierszu – liczba n.*

Przykład

Dla danych wejściowych:

```
12
1 7
7 8
7 11
7 2
2 4
4 10
2 5
5 9
2 6
3 6
3 12
```

jednym z poprawnych wyników jest:

```
1
11
8
7
5
9
2
10
4
6
3
12
```

natomiast dla danych:

```
15
1 14
14 7
7 8
7 11
7 2
2 4
4 10
2 5
5 9
2 6
3 6
3 15
11 12
8 13
```

poprawnym wynikiem jest:

```
BRAK
```

Rozwiązanie

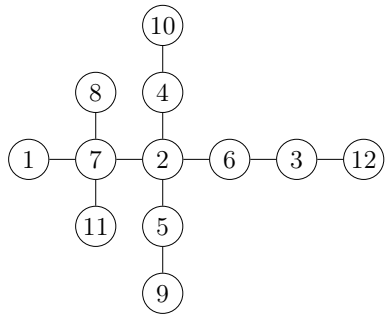
Sieć ulic z treści zadania można opisać jako drzewo, czyli graf o n wierzchołkach oraz $n-1$ krawędziach, spójny i bez cykli. *2-ścieżką* w drzewie nazwiemy ciąg wierzchołków, w którym każde dwa kolejne wierzchołki znajdują się w odległości co najwyżej 2. *2-cyklem* w drzewie nazwiemy 2-ścieżkę zaczynającą się i kończącą w tym samym wierzchołku. Możemy dalej zdefiniować *2H-ścieżkę* (2-ścieżkę Hamiltona) jako 2-ścieżkę przechodzącą przez każdy wierzchołek drzewa dokładnie raz i *2H-cykl* (2-cykl Hamiltona) jako 2-cykl przechodzący przez każdy wierzchołek drzewa dokładnie raz (poza wierzchołkiem początkowym, a zarazem końcowym, cyklu). W naszym zadaniu mamy znaleźć 2H-ścieżkę łączącą dwa zadane wierzchołki drzewa lub stwierdzić, że taka 2H-ścieżka nie istnieje.

Bardzo dawno temu, na II Olimpiadzie Informatycznej [2], pojawiło się zadanie pt. *Obchodzenie drzewa skokami*. W zadaniu tym należało w drzewie znaleźć 3H-cykl,

czyli cykl przechodzący przez każdy wierzchołek drzewa dokładnie raz, w którym każde dwa kolejne wierzchołki są oddalone co najwyżej o 3. Okazywało się wówczas, że każde drzewo zawiera 3H-cykl i istnieje liniowy algorytm, który pozwala wyznaczyć taki 3H-cykl. Nasz przypadek jest jednak bardziej skomplikowany, gdyż nie każde drzewo zawiera 2H-cykl lub nawet 2H-ścieżkę.

Algorytm warstwowy zachłanny

Oznaczmy zbiór wierzchołków drzewa przez $V = \{1, 2, \ldots, n\}$. Niech $1 = u_1$, u_2, …, $u_k = n$ będzie (zwykłą) ścieżką w naszym drzewie łączącą wierzchołki 1 oraz n. Ścieżkę tę będziemy nazywali *ścieżką główną*. Podzielimy wszystkie wierzchołki drzewa na *warstwy* odpowiadające wierzchołkom ścieżki głównej. W i-tej warstwie, oznaczanej jako V_i, znajdą się wszystkie wierzchołki „podłączone do wierzchołka u_i", a więc te wierzchołki, z których ścieżka prowadząca do wierzchołka 1 po raz pierwszy przecina ścieżkę główną właśnie w wierzchołku u_i (patrz rys. 1).



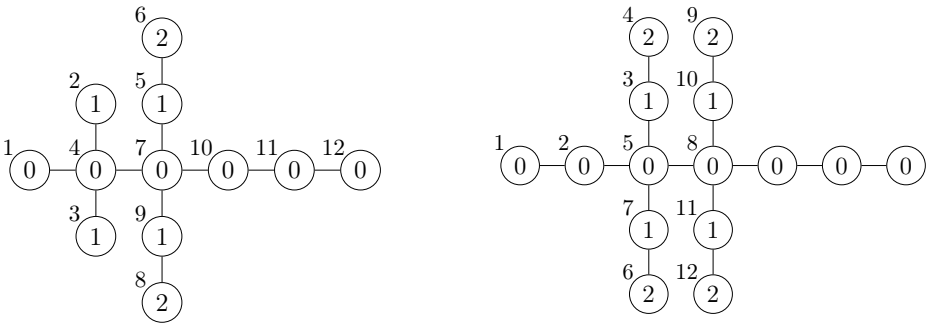
Rys. 1: Inna ilustracja pierwszego przykładu z treści zadania. Ścieżką główną w tym drzewie jest 1, 7, 2, 6, 3, 12, a kolejne warstwy to $\{1\}$, $\{7, 8, 11\}$, $\{2, 4, 5, 9, 10\}$, $\{6\}$, $\{3\}$, $\{12\}$.

Naszą 2H-ścieżkę będziemy konstruowali tak, że najpierw przejdziemy w pewnej kolejności przez wszystkie wierzchołki warstwy V_1, potem przez wszystkie wierzchołki warstwy V_2 itd. Taką 2H-ścieżkę nazwiemy 2H-ścieżką *warstwową*. Poprawność tej decyzji uzasadnia następujący, dość intuicyjny lemat, którego dowód odkładamy na później.

Lemat 1. Jeśli drzewo zawiera 2H-ścieżkę łączącą wierzchołki 1 oraz n, to zawiera także 2H-ścieżkę warstwową łączącą te dwa wierzchołki.

W oparciu o Lemat 1 możemy podać pierwsze rozwiązanie zadania, bazujące na podejściu zachłannym. W rozwiązaniu tym wierzchołki w ramach warstw odwiedzamy zgodnie z priorytetem określonym jako odległość od ścieżki głównej (rys. 2).

```
1:  Algorytm warstwowy zachłanny;
2:  begin
3:    foreach v in V do d(v) := odległość v od ścieżki głównej;
4:    start := 1;
5:    odwiedzaj wierzchołki warstwami, wybierając jako następny wierzchołek
6:      nieodwiedzony wierzchołek o maksymalnym d(v) i stopniu 1, a jeśli takiego
7:      nie ma, dowolny nieodwiedzony wierzchołek o maksymalnym d(v);
8:      { każdy kolejny wierzchołek musi leżeć w odległości 1 lub 2 od bieżącego }
9:    if odwiedziliśmy wszystkie wierzchołki then return true
10:   else return false;
11: end
```



Rys. 2: Priorytety poszczególnych wierzchołków drzew z przykładów w treści zadania (narysowanych warstwowo) oraz 2-ścieżki konstruowane przez algorytm warstwowy zachłanny. W pierwszym przypadku jest to 2H-ścieżka w drzewie, natomiast w drugim przypadku w kroku 12. trafiamy na „pułapkę bez wyjścia" – odwiedzanie kończy się niepowodzeniem, zatem nie ma szukanej 2H-ścieżki.

W implementacji trzeba jeszcze zwrócić uwagę na to, że liczba wierzchołków osiągalnych z jednego wierzchołka w jednym przeskoku może być duża. Aby sobie z tym poradzić, należy scalić każdą grupę liści podłączonych do tego samego wierzchołka w jeden liść. Jeśli po wykonaniu tej operacji jakiś wierzchołek ma stopień większy niż 5, z góry odpowiadamy, że w drzewie nie ma żądanej 2H-ścieżki. Później należy pamiętać, aby w momencie odwiedzania „scalonego" liścia na 2-ścieżce wypisać za jednym zamachem wszystkie liście, z których on powstał.

Okazuje się, że to już całe rozwiązanie. Na dodatek działa ono w czasie liniowym! Jednak nie bardzo widać, dlaczego to rozwiązanie zawsze znajduje 2H-ścieżkę w drzewie, jeśli takowa istnieje. Co więcej, wprowadzone ograniczenie na stopień wierzchołka wygląda co najmniej tajemniczo. Aby wyjaśnić, na jakiej podstawie to rozwiązanie działa, podamy teraz drugie rozwiązanie, oparte na zupełnie innej technice.

Gąsienice i programowanie dynamiczne

Oznaczmy przez $first_i$ pierwszy wierzchołek i-tej warstwy, który odwiedzimy na 2H-ścieżce, a przez $last_i$ – ostatni wierzchołek odwiedzony w tej warstwie. Oczywiście

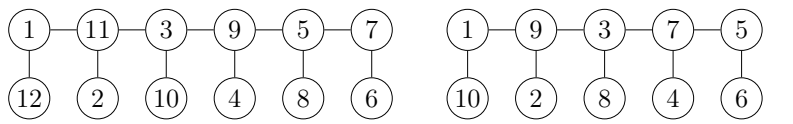
wymagamy, aby $first_1 = 1$ i $last_k = n$. Oznaczmy dalej przez S_i wierzchołek u_i wraz z jego sąsiadami nieleżącymi na ścieżce głównej. Aby dało się przeskoczyć z jednej warstwy bezpośrednio do następnej warstwy, każdy z wierzchołków $last_i$, $first_{i+1}$ musi być albo wierzchołkiem na ścieżce głównej, albo bezpośrednim sąsiadem wierzchołka ze ścieżki głównej, czyli należeć do zbioru S_i lub, odpowiednio, S_{i+1}. W tym rozwiązaniu dla każdego wierzchołka $w \in S_i$ stwierdzimy, czy istnieje 2-ścieżka startująca w wierzchołku 1, przechodząca przez wszystkie wierzchołki warstw $V_1, \ldots, V_i$ i kończąca w wierzchołku $last_i = w$.

Zauważmy, że każde dwa wierzchołki ze zbioru S_i są oddalone co najwyżej o 2. Ponieważ $first_i, last_i \in S_i$, prowadzi nas to do kluczowej obserwacji:

Obserwacja 1. 2H-ścieżka warstwowa ograniczona do jednej warstwy V_i jest 2H-cyklem w tej warstwie.

Powyższa obserwacja wymusza bardzo konkretną postać każdej warstwy. *Gąsienicą* nazwiemy drzewo składające się z jednej ścieżki (tzw. *osi*) oraz z pewnego zbioru wierzchołków (tzw. *odnóg*) „podczepionych" bezpośrednio do wewnętrznych wierzchołków osi. Z danego wierzchołka może wychodzić wiele odnóg, może też nie być do niego podczepiona żadna odnoga. Aby oś gąsienicy była określona jednoznacznie, przyjmujemy, że do końcowych wierzchołków gąsienicy muszą być podłączone jakieś odnogi (poza szczególnym przypadkiem, gdy gąsienica ma tylko jeden wierzchołek).

Jedynymi drzewami posiadającymi 2H-cykl są właśnie gąsienice. Metodę znajdowania 2H-cyklu w gąsienicy przedstawiono na rys. 3. W tym algorytmie, jeżeli jakiejś odnogi nie ma, to omijamy ją bezpośrednim przeskokiem do kolejnego wierzchołka na osi, a jeśli z jednego wierzchołka odchodzi wiele odnóg, odwiedzamy je wszystkie za jednym zamachem. Okazuje się, że jest to jedyny sposób znalezienia 2H-cyklu w gąsienicy. Dowód poniższego lematu również odkładamy na później.



Rys. 3: 2H-cykl w gąsienicy w zależności od parzystości długości osi.

Lemat 2. Drzewo posiada 2H-cykl wtedy i tylko wtedy, gdy jest gąsienicą. Co więcej, każdy 2H-cykl w gąsienicy jest postaci takiej jak na rys. 3.

W ten sposób otrzymujemy następujące rozwiązanie bazujące na metodzie programowania dynamicznego:

1. Wyznacz podział drzewa na warstwy $V_1, \ldots, V_k$.
2. Jeśli któraś z warstw nie jest gąsienicą, wypisz `BRAK`.
3. Dla każdych wierzchołków $v, w \in S_i$ sprawdź, czy gąsienica V_i zawiera 2H-cykl, w którym v i w sąsiadują.

4. Dla każdego $i = 1, \ldots, k$ oraz $w \in S_i$ wyznacz wartość logiczną $Last[w]$:
   (a) Dla każdego $w \in S_1$, $Last[w]$ jest prawdą wtedy i tylko wtedy, gdy V_1 zawiera 2H-cykl łączący 1 oraz w.
   (b) Dla każdego $i > 1$ i $w \in S_i$, $Last[w]$ jest prawdą wtedy i tylko wtedy, gdy istnieją wierzchołki $v \in S_i$ oraz $w' \in S_{i-1}$, takie że $Last[w']$ jest prawdą, wierzchołek v jest osiągalny z w' oraz V_i zawiera 2H-cykl łączący v oraz w.
5. Jeśli $Last[n]$ jest prawdą, odtwórz 2H-ścieżkę, cofając się po wartościach $Last$. W przeciwnym wypadku wypisz `BRAK`.

W powyższym rozwiązaniu $Last[w]$ dla $w \in S_i$ jest prawdą wtedy i tylko wtedy, gdy istnieje 2-ścieżka warstwowa startująca w wierzchołku 1, przechodząca przez wszystkie wierzchołki warstw $V_1, \ldots, V_i$ i kończąca w wierzchołku $last_i = w$. Czyli jest to dokładnie to, o co nam chodziło!

Rozwiązanie miałoby złożoność liniową, gdyby nie fakt, że zbiory S_i mogą być duże. Jednak w zbiorze S_i mogą znajdować się co najwyżej trzy wierzchołki niebędące liśćmi drzewa – w przeciwnym razie i-ta warstwa nie byłaby gąsienicą. Zauważmy, że wszystkie liście w zbiorze S_i są dla nas takie same. Stąd w powyższym algorytmie wystarczy w zbiorze S_i pozostawić maksymalnie dwa liście (jako kandydatów na wierzchołki v i w). W ten sposób S_i nie będzie miał nigdy więcej niż pięć elementów. Dzięki temu w każdej warstwie będziemy poszukiwać 2H-cyklu tylko stałą liczbę razy. Ponadto obliczenie $Last[w]$ będzie wymagało sprawdzenia $Last[w']$ tylko dla stałej liczby wierzchołków w'.

Ostatecznie całe rozwiązanie działa w czasie liniowym. Implementację rozwiązania opartego na programowaniu dynamicznym można znaleźć w pliku `mul.cpp`, natomiast implementację algorytmu warstwowego zachłannego – w pliku `mul2.pas`. Poniżej uzupełniamy brakujące dowody lematów.

Dowód lematu 1

Jeśli P jest 2H-ścieżką z 1 do n w drzewie, to przez $ind(P)$ oznaczymy najmniejsze $i \in \{1, \ldots, k-1\}$, takie że istnieje wierzchołek z warstwy V_i leżący później na P niż jakiś wierzchołek z dalszej warstwy (tj. warstwy V_j dla $j > i$). Jeśli taki indeks i nie istnieje, przyjmujemy $ind(P) = k$. Na mocy założeń lematu wiemy, że istnieje 2H-ścieżka prowadząca z 1 do n. Niech P oznacza 2H-ścieżkę maksymalizującą wartość $ind(P)$. Wykażemy, że P jest 2H-ścieżką warstwową.

Oczywiście jeśli $ind(P) = k$ to P jest warstwowa. Przyjmijmy zatem, że $ind(P) = i < k$. Udowodnimy, że prowadzi to do sprzeczności. Podzielmy P na fragmenty $X_1, X_2, X_3, \ldots, X_l$ w taki sposób, żeby każdy wierzchołek fragmentu typu X_{2p-1} (dla $p = 1, 2, \ldots$) należał do zbioru $\mathcal{L} = \bigcup_{j \leqslant i} V_j$, a każdy wierzchołek fragmentu typu X_{2p} (dla $p = 1, 2, \ldots$) należał do zbioru $\mathcal{R} = \bigcup_{j > i} V_j$. Na mocy założenia mamy, że $l \geqslant 4$.

Oznaczmy przez $N(v)$ wierzchołek v wraz z jego sąsiadami. Zauważmy, że

$$first(X_{2p-1}),\ last(X_{2p-1}) \quad \in \quad \{u_1\} \cup N(u_i)$$

$$\mathit{first}(X_{2p}),\ \mathit{last}(X_{2p}) \quad \in \quad \{u_k\} \cup N(u_{i+1}).$$

Ponadto, dla każdego $j = 1, \ldots, l-1$ musi zachodzić $\mathit{last}(X_j) \in \{u_i, u_{i+1}\}$ lub $\mathit{first}(X_{j+1}) \in \{u_i, u_{i+1}\}$.

Po chwili namysłu można dojść do wniosku, że jedyną możliwością spełnienia powyższych warunków jest sytuacja, gdy $l = 4$ oraz:

$$\mathit{last}(X_1) \in N(u_i) \setminus \{u_i\}, \quad \mathit{first}(X_2) = u_{i+1}, \quad \mathit{last}(X_2) \in N(u_{i+1}),$$

$$\mathit{first}(X_3) \in N(u_i), \quad \mathit{last}(X_3) = u_i, \quad \mathit{first}(X_4) \in N(u_{i+1}) \setminus \{u_{i+1}\}.$$

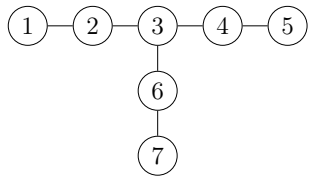
Sprawdźmy, że tak rzeczywiście jest. Ponieważ każda para ścieżek X_{2p-1}, X_{2p} zawiera jakiś wierzchołek spośród $\{u_i, u_{i+1}\}$, a l jest parzyste, więc l nie może być większe niż 4. Stąd $l = 4$.

Załóżmy teraz, że zachodziłoby $\mathit{last}(X_1) = u_i$. Wówczas $\mathit{last}(X_3) \neq u_i$, a zatem $\mathit{first}(X_4) = u_{i+1}$. Ponadto $\mathit{first}(X_3) \neq u_i$, więc także $\mathit{last}(X_2) = u_{i+1}$, co prowadzi do sprzeczności. W ten sposób wykazaliśmy, że $\mathit{last}(X_1) \neq u_i$, skąd otrzymujemy, że $\mathit{last}(X_1) \in N(u_i) \setminus \{u_i\}$ oraz $\mathit{first}(X_2) = u_{i+1}$. Musi dalej zachodzić $\mathit{last}(X_3) = u_i$, gdyż w przeciwnym razie mielibyśmy $\mathit{first}(X_4) = u_{i+1}$, co nie jest możliwe. W ten sposób otrzymujemy, że rzeczywiście P ma taką strukturę, jak opisana powyżej. (Można zauważyć, że struktura ta na styku X_2 i X_3 wymusza, że $X_2 = \{u_{i+1}\}$ lub $X_3 = \{u_i\}$, jednak to spostrzeżenie nie będzie nam potrzebne).

W tej sytuacji $P' = X_1 X_3 X_2 X_4$ także jest 2H-ścieżką w drzewie prowadzącą z 1 do n. Zauważmy, że kolejność występowania wierzchołków ze zbioru $\mathcal{L}$ na ścieżkach P i P' jest taka sama. Co więcej, na ścieżce P' wszystkie wierzchołki ze zbioru $\mathcal{R}$ występują później niż wszystkie wierzchołki ze zbioru $\mathcal{L}$. Stąd $\mathit{ind}(P') > i = \mathit{ind}(P)$, co daje żądaną sprzeczność wobec metody wyboru ścieżki P.

Dowód lematu 2

Zacznijmy od uzasadnienia stwierdzenia, jeśli drzewo ma 2H-cykl, to musi ono być gąsienicą. Przyjrzyjmy się prostemu drzewu z rys. 4 (trzy przedłużone odnogi). Łatwo zauważyć, że drzewo jest gąsienicą wtedy i tylko wtedy, gdy nie zawiera drzewa z rys. 4 jako poddrzewa. Na mocy poniższego lematu o przycinaniu, gdyby jakieś drzewo niebędące gąsienicą miało 2H-cykl, to również drzewo z rys. 4 miałoby 2H-cykl.



Rys. 4: Najprostsze drzewo niezawierające 2H-cyklu.

Lemat 3 (O przycinaniu). *Niech T będzie drzewem o co najmniej dwóch wierzchołkach, w – liściem drzewa T, a S – dowolną 2-ścieżką w T. Wówczas $S \setminus \{w\}$ (tj. S z usuniętymi wystąpieniami wierzchołka w) jest 2-ścieżką w drzewie $T \setminus \{w\}$.*

Dowód: Na 2-ścieżce S wierzchołek w może sąsiadować jedynie ze swoim bezpośrednim sąsiadem v w T oraz z sąsiadami tego sąsiada. Zauważmy, że każdy wierzchołek ze zbioru $N(v)$ (v oraz jego sąsiedzi) jest odległy od każdego innego wierzchołka tego zbioru co najwyżej o 2. To oznacza, że po usunięciu wszystkich wystąpień w z S dalej mamy 2-ścieżkę. ■

Łatwo sprawdzić, że drzewo z rys. 4 nie zawiera 2H-cyklu. Stąd rzeczywiście jedynymi drzewami mogącymi mieć 2H-cykl są gąsienice.

Kolejnym krokiem jest formalne uzasadnienie faktu, że każda gąsienica zawiera 2H-cykl. Najlepiej zacząć od znalezienia 2H-cyklu na osi gąsienicy. Łatwo podać ogólną postać takiego 2H-cyklu, zależnie od parzystości długości osi: startujemy z jednego końca osi, skaczemy co drugi wierzchołek do drugiego końca osi, po czym zawracamy i znów skaczemy co dwa (rys. 5). Co więcej, widzimy, że na danej osi jest dokładnie jeden 2H-cykl (z dokładnością do wyboru wierzchołka początkowego i kierunku obejścia).

Rys. 5: 2H-cykl na osi gąsienicy.

Jeśli gąsienica zawiera odnogi, wystarczy umieścić je w odpowiednim miejscu opisanego wyżej 2H-cyklu. Załóżmy, że osią gąsienicy jest ścieżka $v_1, \ldots, v_l$. Wówczas odnogi podłączone do wierzchołka v_i (gdzie $1 < i < l$) należy umieścić na 2H-cyklu między wystąpieniami wierzchołków v_{i-1} i v_{i+1}, natomiast odnogi wychodzące z wierzchołka v_1 (odpowiednio v_l) umieścić należy między wystąpieniami wierzchołków v_1 oraz v_2 (odpowiednio v_{l-1} oraz v_l). W ten sposób otrzymamy dokładnie taki 2H-cykl, jak na przedstawionym wcześniej rysunku 3.

Ostatnią rzeczą, jaka pozostała nam do uzasadnienia, jest fakt, że każdy 2H-cykl w gąsienicy jest dokładnie opisanej postaci. Dokładniej, poszczególne 2H-cykle w danej gąsienicy różnią się tylko kolejnością odwiedzania odnóg podłączonych do tego samego wierzchołka osi (a także, rzecz jasna, wyborem wierzchołka początkowego 2H-cyklu i kierunku obejścia).

W uzasadnieniu możemy bazować na poczynionym już spostrzeżeniu, iż 2H-cykl na osi gąsienicy jest określony jednoznacznie. Na mocy lematu o przycinaniu, 2H-cykl w dowolnej gąsienicy ograniczony do wierzchołków jej osi jest 2H-cyklem tej osi. Każda z odnóg wpisuje się zatem w którymś miejscu jedynego możliwego 2H-cyklu osi gąsienicy. Wystarczy uzasadnić, że miejsce to jest wyznaczone jednoznacznie. Rozważmy zatem odnogę podłączoną do wierzchołka v_i, dla $2 < i < l - 1$. Wówczas mogłaby ona zostać dołączona albo na fragmencie $\ldots, v_{i-1}, v_{i+1}, \ldots$, albo na fragmencie $\ldots, v_{i+2}, v_i, v_{i-2}, \ldots$ Widzimy, że nie zaburzy ona własności 2H-cyklu tylko wtedy, gdy umieścimy ją na 2H-cyklu między wierzchołkami v_{i-1} oraz v_{i+1}. Podobnie można sprawdzić, że umiejscowienie odnóg podłączonych do wierzchołków v_1, v_2, v_{l-1} oraz v_l w ramach 2H-cyklu osi gąsienicy jest wyznaczone jednoznacznie. Ten szczegół dowodu pozostawimy Czytelnikowi jako ćwiczenie.